Algebraic Structures for Capturing the Provenance of SPARQL Queries

Floris Geerts, Thomas Unger, Grigoris Karvounarakis, Irini Fundulaki, Vassilis Christophides

Algebraic Structures for Capturing the Provenance of SPARQL Queries

FLORIS GEERTS, University of Antwerp
THOMAS UNGER, University College Dublin
GRIGORIS KARVOUNARAKIS, LogicBlox
IRINI FUNDULAKI, ICS-FORTH
VASSILIS CHRISTOPHIDES, Inria Paris-Rocquencourt

The evaluation of SPARQL algebra queries on various kinds of annotated RDF graphs can be seen as a particular case of the evaluation of these queries on RDF graphs annotated with elements of so-called *spm-semirings*. Spm-semirings extend semirings, used for representing the provenance of positive relational algebra queries on annotated relational data, with a new operator to capture the semantics of the non-monotone SPARQL operators. Furthermore, spm-semiring-based annotations ensure that desired SPARQL query equivalences hold when querying annotated RDF. In this work, in addition to introducing spm-semirings, we study their properties and provide an alternative characterization of these structures in terms of semirings with an embedded boolean algebra (or seba-structure for short). This characterization allows us to construct spm-semirings and identify a universal object in the class of spm-semirings. Finally, we show that this universal object provides a provenance representation of poly-sized overhead and can be used to evaluate SPARQL queries on arbitrary spm-semiring-annotated RDF graphs.

CCS Concepts: • **Information systems → Data provenance; Query languages for non-relational engines**

Additional Key Words and Phrases: Algebraic structures, annotations, provenance models, query languages, RDF, SPARQL

ACM Reference Format:
Floris Geerts, Thomas Unger, Grigoris Karvounarakis, Irini Fundulaki, and Vassilis Christophides. 2016. Algebraic structures for capturing the provenance of SPARQL queries. J. ACM 63, 1, Article 7 (February 2016), 63 pages.
DOI: http://dx.doi.org/10.1145/2810037

1. INTRODUCTION

The W3C Linked Data Initiative has boosted the publication and interlinkage of massive amounts of scientific, corporate, governmental, and crowd-sourced datasets on the emerging Data Web. These data are commonly published in the form of RDF data [Manola et al. 2004] and queried with the SPARQL query language [Prud'hommeaux

Authors' addresses: F. Geerts, Department of Mathematics & Computer Science, University of Antwerp, Middelheimlaan 1, 2020 Antwerp, Belgium; email: floris.geerts@uantwerpen.be; T. Unger, School of Mathematics and Statistics, University College Dublin, Belfield, Dublin 4, Ireland; email: thomas.unger@ucd.ie; G. Karvounarakis, LogicBlox, 1349 West Peachtree Street NW, Suite 1880, Atlanta, GA 30309, USA; email: grigoris.karvounarakis@logicblox.com; I. Fundulaki, Institute of Computer Science (ICS), Foundation for Research and Technology-Hellas (FORTH), N. Plastira 100, Vasilika Vouton, Heraklion, Greece, 70013; email: fundul@ics.forth.gr; V. Christophides, Inria Paris-Rocquencourt. Laboratory of Information, Networking and Communication Sciences, 23 avenue d'Italie, 75013 Paris, France; email: Vassilis.Christophides@inria.fr.

DOI: http://dx.doi.org/10.1145/2810037

and Seaborne 2008]. In such settings where RDF data are freely exchanged, integrated, and materialized in distributed repositories, it is crucial to be able to assess the quality of replicated and possibly incomplete or uncertain data.

Toward this end, several models for annotated RDF data have been proposed [Carroll et al. 2005; Mazzieri and Dragoni 2008; Hartig 2009; Dividino et al. 2009; Huang and Liu 2009; Zimmermann et al. 2012; Straccia 2013] in an attempt to represent various dimensions of data quality such as *trust*, *truth* of imprecise information, or the *probability* of the validity of the data. In all these cases, when annotated data are transformed through SPARQL queries, one needs to compute appropriate annotations for the query results. For instance, in the case of *trust assessment* [Hartig 2008, 2009; Dividino et al. 2009], the trustworthiness of query results is determined based on the trustworthiness of source datasets from which they were derived. Similarly, for *uncertain* and *fuzzy* datasets, the degrees of truth of query results are derived based on the degrees of truth associated with the original data [Dividino et al. 2009; Huang and Liu 2009; Zimmermann et al. 2012; Straccia 2013].

If we are only interested in one kind of annotations, and source annotations are static and common for all users as well as available at query evaluation time, such computations can be performed during query evaluation [Hartig 2008, 2009; Dividino et al. 2009; Huang and Liu 2009; Zimmermann et al. 2012; Straccia 2013].

In general however, various different scenarios may present themselves: Different applications may require computing different kinds of annotations over the same source data and queries; for a single kind of annotation (especially in data integration and warehouse settings) data may be collected from various RDF sources that change over time; different users may have different beliefs about aspects of the data and these beliefs may not be available at query evaluation time or may change over time; different users may only be interested in computing annotations for a small subset of the results of a query, and the like.

As a consequence, we may end up with redundant query evaluations. The same query may have to be evaluated repeatedly over the same source data for each kind of annotation. Even if source annotations representing the beliefs of each user are available at query evaluation time, the same query may have to be evaluated separately for each one of these sets of source annotations despite the fact that the relationship of query results with the source data is the same for all users. Because we do not know in advance which query results will be derived, or which subset of the source data is involved in their derivation, annotations for all results of each query may need to be computed during query evaluation.

For these reasons, *abstract provenance models* [Green et al. 2007] have been introduced: They use *abstract tokens* to represent tuple annotations and *abstract operators* to capture the relationship between the query operators that combine source data to derive query results. Conceptually, the resulting *abstract provenance expressions* encode for each query result its relationship to source data and query operators, as implied by the structure of the query, independently of any specific kind of annotations or particular source data annotation values. Such expressions can then be computed once during query evaluation, and annotations for various applications or users with different beliefs or specific query results can be computed from them, without requiring redundant re-evaluation(s) of each query over all source data.

In the relational setting, provenance models that are capable of abstracting the query evaluation on annotated relational data have been put forward for the positive fragment of the relational algebra [Cheney et al. 2009]. In particular, the modeling of annotations by means of *semirings* has shown great promise [Green et al. 2007], both as a platform for theoretical study and as a representation employed in systems that record provenance information when data are imported in the hosting repository

[Karvounarakis et al. 2013] and use it to compute appropriate annotations for different applications and users at a later time [Karvounarakis et al. 2010].

For annotated RDF data and *positive* SPARQL queries (those that use only the AND and UNION operators), one can verify, similarly to the positive relational case, that semirings suffice as the annotation structure [Theoharis et al. 2011]. However, when the *non-monotone* SPARQL operator OPTIONAL is brought into the picture, it can easily be verified that extensions of semirings have to be considered. Indeed, the SPARQL 1.0 algebra semantics [Pérez et al. 2009] of OPTIONAL is defined in terms of a left-outer join that involves a non-monotone *difference* algebraic operator. More specifically, for any two RDF data sets G_1 and G_2, (G_1 OPTIONAL G_2) can be written as (G_1 AND G_2) UNION (G_1 DIFFERENCE G_2) [Pérez et al. 2009]. Although DIFFERENCE is—strictly speaking—not part of the SPARQL 1.0 specification, we add it for convenience.[1]

However, as explained in Section 2, the semantics of DIFFERENCE and OPTIONAL are not the same as those of relational difference and (left-)outer join, respectively. Thus, extensions to the semiring framework intended to cope with the latter [Geerts and Poggi 2010; Amsterdamer et al. 2011a, 2011b; Glavic and Alonso 2009] cannot be applied directly to capture the provenance of SPARQL queries. Recent work [Damásio et al. 2012] suggests that it may be possible to express the SPARQL difference in terms of relational operators to leverage the structure of *m*-semirings, an extension of semirings for relational queries involving difference [Geerts and Poggi 2010]. However, whereas a provenance model based on the so-called universal *m*-semiring exists, this model does not allow for a concise and simple representation of its expressions [Geerts and Poggi 2010] and, thus, its practical usability as a provenance model is rather limited.

For these reasons, we propose a new algebraic structure for capturing the semantics of SPARQL DIFFERENCE (and thus also OPTIONAL) in this article. More specifically, we identify a set of SPARQL query equivalences that involve DIFFERENCE and hold under both bag and set semantics, and we show that these equivalences also hold when evaluating SPARQL queries on a wide variety of annotated RDF data. Then, we define so-called *spm-semirings*, an extension of semirings with a new operation $\ominus$, based on identities derived from the aforementioned SPARQL equivalences. Furthermore, we show that spm-semirings do have a universal structure that provides a concise representation of the provenance of RDF data and SPARQL queries involved.

The underlying techniques rely on a characterization of spm-semirings in terms of semirings with an embedded boolean algebra, or *seba-structure* for short. This characterization is nontrivial and may be of interest in its own right. Furthermore, the spm-semiring-based provenance expressions can indeed be used to compute appropriate annotations in a wide variety of application domains. We thus provide a complete picture of SPARQL query evaluation on annotated RDF and propose an abstract provenance model that incorporates non-monotone SPARQL operators. We note that the relational analogue is still open for relational queries with difference.

In summary, we make the following contributions:

(1) We illustrate that the semantics of SPARQL on various notions of annotated RDF have a great commonality (Section 2). Based on this, we generalize the semantics of SPARQL algebra expressions to RDF data annotated with values from some arbitrary annotation domain K, or K-annotated RDF for short (Section 3). For this purpose, K is equipped with binary operations $\oplus$, $\otimes$, and $\ominus$, and constants 0 and 1 that accommodate all SPARQL algebra operators.

[1]The algebra described in the SPARQL 1.1 specification [Harris and Seaborne 2013] also contains a similar operator called MINUS. We discuss SPARQL 1.1 in Section 8.

(2) We identify a set of SPARQL algebra equivalences (some involving DIFFERENCE) that are desirable to hold on K-annotated RDF (Section 4). We show that for these equivalences to hold, $(K, \oplus, \otimes, \ominus, 0, 1)$ must be an *spm-semiring* and vice versa. A minimal set of identities defining spm-semirings is provided.
(3) An alternative characterization of spm-semirings is given in Section 5, based on semirings with an embedded boolean algebra (seba-structures). We prove the correctness of this characterization and show how it can be used to construct spm-semirings based on semirings commonly used in practice.
(4) We identify a universal object in the class of spm-semirings (Section 6), leveraging the characterization in terms of seba-structures, and show that the evaluation of SPARQL queries on spm-semiring annotated RDF factors through the evaluation of RDF annotated with elements in the universal object. The universal object is therefore proposed in Section 7 as a provenance model for annotated RDF and SPARQL. Furthermore, we explain how newly introduced non-monotone operators in SPARQL 1.1 fit into our algebraic framework in Section 8. Finally, we compare spm-semirings with related work in the relational and semantic Web contexts in Section 9 and conclude with directions for future work in Section 10.

This article is a considerable expansion of the 12-page conference paper [Geerts et al. 2013]. Apart from providing all definitions and proofs, we have included ample examples illustrating key concepts. Furthermore, we have corrected several mistakes, especially those concerning the construction of the universal objects.

2. QUERYING ANNOTATED RDF

In this section, we provide examples of the evaluation of SPARQL queries on annotated RDF data. We first recall RDF [Manola et al. 2004] and SPARQL [Prud'hommeaux and Seaborne 2008] with the standard bag semantics in which we represent multiplicities as annotations. We then observe that, similar to the relational case, the semantics of SPARQL on various forms of annotated RDF have a great commonality.

2.1. RDF and SPARQL in a Nutshell

RDF is the standard model for representing semantic Web data as sets of *triples* of the form (*subject*, *predicate*, *object*). Intuitively, for each triple, the *predicate* describes the relationship between *subject* and *object*. For instance, Table (a) of Figure 1 shows an example of an RDF triple set, denoted by G, where the columns stand for the corresponding components of each triple (**S** for *subject*, **P** for *predicate,* and **O** for *object*). SPARQL is the standard language used to query RDF data. We present the SPARQL semantics based on the algebra of Pérez et al. [2009]. The operators of this algebra manipulate bags of mappings (i.e., valuations of variables to constants in G) and include unary operators σ and π that correspond to the SPARQL constructs FILTER and SELECT, respectively, and binary operators $\cup$, $\bowtie$, and $\ltimes$ for the SPARQL constructs UNION, AND, and OPTIONAL, respectively. The operator $\ltimes$ can be defined in terms of $\cup$, $\bowtie$, and $\setminus$, the algebraic counterpart of DIFFERENCE [Pérez et al. 2009]. The following example illustrates the standard bag semantics of SPARQL queries.

Example 2.1 (*bags*). In Figure 1 we start by considering the simplest SPARQL query $(?x, ?y, ?z)$ over the RDF triple set G depicted in Table (a). Such a query is referred to as a triple pattern where $?x$, $?y$, and $?z$ denote variables. Intuitively, the evaluation of the triple pattern $(?x, ?y, ?z)$ consists of a bag of mappings. Each mapping corresponds to the selection of a triple from the RDF triple set G by bounding the variables to constants in triples in G. Table (b) shows the resulting mapping bag Ω. We employ

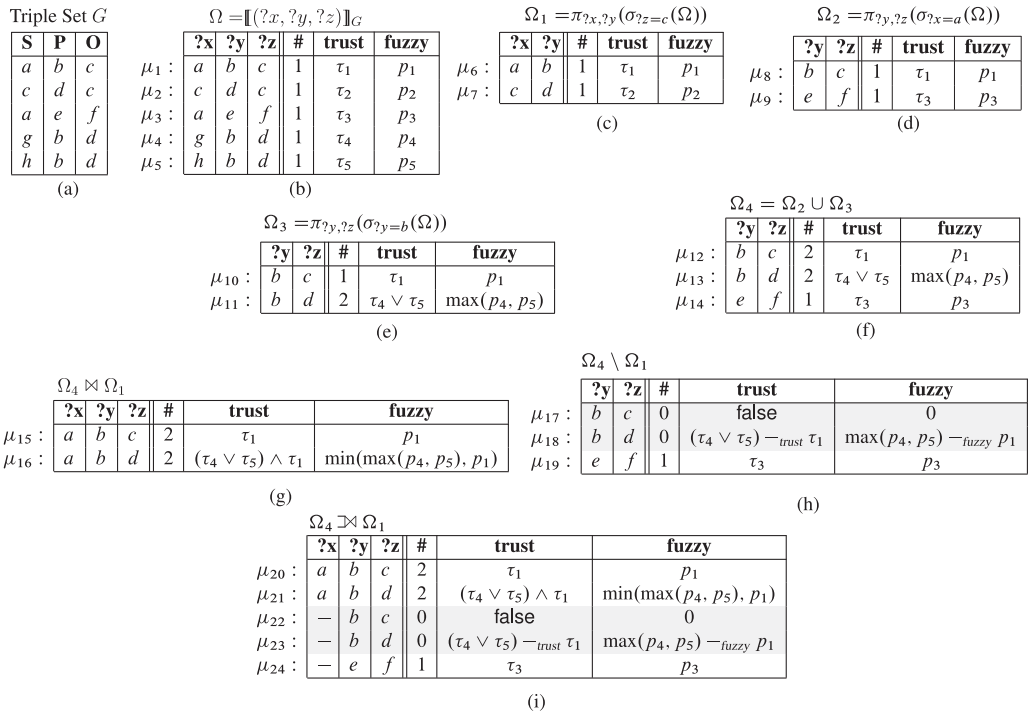

Fig. 1. Example of RDF graph and evaluation of SPARQL algebra operators (bag/trust/fuzzy semantics).

symbol μ_i to identify individual mappings. The formal definition of triple patterns and general SPARQL queries is provided at the beginning of Section 3.

To simplify the presentation, we use the tabular representation of the mapping bags shown in Table (b) in Figure 1, where the first three columns correspond to variables in the mappings and the fourth column (**#**) represents the multiplicity of the mapping. The last two columns (**trust** and **fuzzy**) can be ignored for now, as can the gray shaded entries in Tables (h) and (i).

Tables (c–e) illustrate the evaluation of the operators σ and π. The output mapping bags are denoted by Ω_1, Ω_2, and Ω_3. For instance, mapping μ_{11} of Ω_3 has two derivations originating from mappings in Ω; namely, one by projecting μ_4 and another one by projecting μ_5. The multiplicity of μ_{11} is obtained by *adding* the multiplicities of μ_4 and μ_5.

Table (f) shows the result of $\Omega_4 = \Omega_2 \cup \Omega_3$, where Ω_2 and Ω_3 are shown in Tables (d) and (e), respectively. In Ω_4, the mapping μ_{13} has two derivations, both of them originating from Ω_3, whereas the two derivations of μ_{12} originate from Ω_2 (μ_8) and Ω_3 (μ_{10}). The multiplicity of μ_{12} is obtained by *adding* the multiplicities of μ_8 and μ_{10}.

Table (g) depicts the result of $\Omega_4 \bowtie \Omega_1$. For instance, mapping μ_{16} is derived by joining $\mu_{13} \in \Omega_4$ and $\mu_6 \in \Omega_1$. Mappings can be joined only if they are *compatible* [Pérez et al. 2009]. In our example, μ_6 and μ_{13} are compatible because they agree on their common variable (i.e., they both bind $?y$ to b). The multiplicity of μ_{16} is computed as the *product* of the multiplicities of the two input mappings ($1 \times 2 = 2$).

Table (h) illustrates an example of the difference operator (i.e., $\Omega_4 \setminus \Omega_1$). Note that neither μ_{12} nor μ_{13} in Ω_4 are in the result—recall that we ignore the gray-shaded entries for now—because Ω_1 contains a mapping (μ_6) that is compatible with the mappings μ_{12} and μ_{13} in Ω_4 as explained earlier. On the other hand, there is no mapping in

Ω_1 that is compatible with μ_{14}. As a consequence, μ_{14} appears in the result as μ_{19} in Table (h). Note that the difference operator can be applied on bags of mappings defined on different variables since only the common variables of the mappings are considered. This is also true for the SPARQL union.

Finally, Table (i) depicts the result of $\Omega_4 \mathbin{⟕} \Omega_1$, which contains all mappings from Tables (g) and (h). Indeed, $\Omega_4 \mathbin{⟕} \Omega_1 = (\Omega_4 \bowtie \Omega_1) \cup (\Omega_4 \setminus \Omega_1)$. The symbol "–" denotes that $?x$ is not bound to a constant in the mappings μ_{22}, μ_{23} and μ_{24}.

The previous example shows that the multiplicities of mappings in the result of σ, π, $\cup$, and $\bowtie$ SPARQL algebra operators are computed in a similar way as when evaluating the corresponding operators of the relational algebra under bag semantics [Green et al. 2007]. In particular, in the case of alternative derivations (e.g., for π or $\cup$) of a mapping, its multiplicity equals the sum of the multiplicities of the different derivations. In the case of $\bowtie$, the multiplicity of the result mapping equals the product of the multiplicities of the two input mappings that were combined.

However, the multiplicities of the result mappings of the $\setminus$ operator are computed differently from the corresponding case of the difference operator ($\setminus_{ra}$) in the relational algebra under bag semantics. Indeed, let R and S be two relations and denote by $R(t)$ and $S(t)$ the multiplicity of a tuple t in R and S, respectively. Then, $(R \setminus_{ra} S)(t) := \max(0, R(t) - S(t))$ in the bag semantics of the relational algebra [Green et al. 2007]. In contrast, the SPARQL difference ($\setminus$) is defined in terms of compatibility, not equality [Schmidt et al. 2010]. Indeed, Table (h) shows that when considering the SPARQL difference $\Omega_4 \setminus \Omega_1$, a mapping (tuple) t in Ω_4 is in the output as long as there is no mapping in Ω_1 that is compatible with it. That is, $(\Omega_4 \setminus \Omega_1)(t) = \Omega_4(t) -_{bag} \Sigma_{t' \sim t} \Omega_1(t')$, where for any two natural number n and m, $n -_{bag} m = n$ in case that $m = 0$, and $n -_{bag} m = 0$ otherwise. Thus, SPARQL follows the standard relational bag semantics for its positive operators but uses a different semantics for $\setminus$. Similarly, relational left-outer join, which is defined as a union between a join and a relational difference, also uses a different semantics than SPARQL `OPTIONAL`.

Example 2.2 (*bags cont'd*). Consider Tables (c), (f), and (h) of Figure 1. Observe that although μ_{12} and μ_{13} both have multiplicity 2 in Ω_4 (Table (f)), and the compatible mappings μ_6 and μ_7 in Ω_1 have multiplicity 1 (Table (c)), neither μ_{12} nor μ_{13} is present in $\Omega_4 \setminus \Omega_1$. As another example, consider $\pi_{?y}(\Omega_1)$ consisting of mappings μ'_6 and μ'_7 obtained as the projection on the variable $?y$ of μ_6 and μ_7 in Ω_1, respectively. These mappings both have multiplicity 1. Furthermore, consider $\pi_{?y}(\Omega_4)$ consisting of mappings μ'_{12+13} and μ'_{14} obtained as the projection on the variable $?y$ of μ_{12} and μ_{13}, and μ_{14} in Ω_4, respectively. The mapping μ'_{12+13} has multiplicity 4 and the mapping μ'_{14} has multiplicity 1. As before, when considering $\pi_{?y}(\Omega_4) \setminus \pi_{?y}(\Omega_1)$, the mapping μ'_{12+13} will not be present in the result despite the fact that μ'_{12+13} has a higher multiplicity than μ'_6. Since the mapping bags $\pi_{?y}(\Omega_1)$ and $\pi_{?y}(\Omega_4)$ are defined over the same schema (with "attribute" $?y$), we can also consider $\big(\pi_{?y}(\Omega_4)\big) \setminus_{ra} \big(\pi_{?y}(\Omega_1)\big)$. In this case, μ'_{12+13} would be in the result with multiplicity 3, as given by $\max(0, 4 - 1)$.

2.2. SPARQL on Annotated RDF

We next consider the semantics of SPARQL when RDF is adorned with trust information [Hartig 2008, 2009; Dividino et al. 2009]. In this setting, for a given SPARQL query, the goal is to find which result mappings are trusted based on the trustworthiness of the input mappings. More specifically, in case of mappings with multiple derivations, a single trusted derivation suffices to infer that the result mapping is trusted. When two mappings are combined in a derivation, both of them should be trusted in order for the result mapping to be trusted. Based on this semantics, which has also been

studied in the relational context [Karvounarakis et al. 2013; Green et al. 2007], one can compute the trusted result mappings by answering the query over the subset of the input consisting of trusted triples only. This semantics also coincides with the set semantics of SPARQL [Pérez et al. 2009] in which a trusted mapping belongs to the output mapping set and an untrusted mapping does not.

Example 2.3 (*trust, set*). Consider Figure 1 where, instead of the **#** column, we now focus on the annotations in the **trust** column. For example, each triple in Table (b) comes with a boolean trust value (τ_i) that is true if the triple is trusted and false otherwise. It is readily verified that the desired trust semantics is obtained for the example SPARQL queries in Figure 1 by combining trust values through *disjunction* ($\vee$) and *conjunction* ($\wedge$), instead of addition and multiplication, respectively, used to compute multiplicities in Example 2.1. For example, μ_{11} in Table (e) is trusted only if one of the mappings μ_4 or μ_5 is trusted. Similarly, μ_{16} in Table (g) is trusted if μ_1 is trusted *and* either μ_4 or μ_5 is trusted. To deal with \, we consider boolean negation (denoted by $\bar{\tau}$ for a trust variable τ). Indeed, consider the gray-shaded entry μ_{18} in Table (h). This mapping is trusted only if μ_1 is *untrusted* and either μ_4 or μ_5 is trusted. This is expressed by $(\tau_4 \vee \tau_5) \wedge \bar{\tau}_1$ and denoted by $(\tau_4 \vee \tau_5) -_{trust} \tau_1$ in Table (h). Hence, the gray-shaded mappings can be part of the result depending on the trust information of the source mappings. In general, if φ and ψ are two propositional formulas over boolean variables, then their difference is defined as $\varphi -_{trust} \psi := \varphi \wedge \bar{\psi}$. Note that this is similar to the notion of difference given in the bag semantics (cf. Example 2.1): $\varphi \wedge \bar{\psi}$ equals φ if ψ is false, and equals false otherwise.

We conclude this section by considering SPARQL on fuzzy RDF data [Zimmermann et al. 2012; Straccia 2013]. In this setting, every mapping is annotated with a real number in the range $[0, 1]$, where the annotation denotes the degree of truth that the mapping exists in the particular mapping set. Mappings annotated with 0 certainly do not exist in the mapping set, whereas those annotated with 1 certainly exist. In the case of mappings with alternative derivations, the degree of truth of the mapping is that of the derivation with the highest degree of truth while the degree of truth of a composite derivation equals the minimum of the degrees of truth over the two input mappings.

Example 2.4 (*fuzzy*). Consider Figure 1 where instead of the **#** column, we now focus on the **fuzzy** column. For example, each triple in Table (b) comes with a degree of truth (p_i) that is a real number in $[0, 1]$. It is readily verified that the desired fuzzy semantics is obtained for the example SPARQL queries in Figure 1 if we take the *maximum* (max) and *minimum* (min) instead of addition and multiplication, respectively, as used for bag semantics (cf. Example 2.1). To deal with \, we need to consider an additional operator $-_{fuzzy}$ on degrees of truth that is defined as $p -_{fuzzy} q := p$ in case that $q = 0$, and $p -_{fuzzy} q := 0$ otherwise. This definition is in line with the treatment of the `OPTIONAL` operator in the fuzzy RDF setting [Zimmermann et al. 2012; Straccia 2013]. For example, the gray-shaded entry μ_{18} in Table (h) appears with degree of truth $\max(p_4, p_5) -_{fuzzy} p_1$. That is, if μ_1 has non-zero degree of truth, then μ_{18} will not appear in the query result. Similarly, if both p_4 and p_5 have zero degree of truth, then μ_{18} is not part of the output. In all other cases, μ_{18} is a result mapping with degree of truth $\max(p_4, p_5)$. Note again the similarity between $-_{fuzzy}$, $-_{bag}$ and $-_{trust}$.

We remark that Dividino et al. [2009] define $p -_{fuzzy} q$ as $\min(p, 1 - q)$. Although this definition collapses to $-_{trust}$ when p and q can only be 0 or 1, it is not compatible with the standard bag semantics. Instead, we aim to provide a general treatment that covers the SPARQL semantics in the setting of sets and bags and beyond. Furthermore,

defining $p -_{fuzzy} q$ as in Dividino et al. [2009] does lead to a semantics of SPARQL for which desired query equivalences (introduced in the next section) are not satisfied. We therefore adopt the fuzzy semantics given by Zimmermann et al. [2012] and Straccia [2013].

2.3. Summary and Lookahead

The previous examples suggest a commonality between the different semantics of SPARQL on annotated RDF. Similar to the semiring-based approach for annotated relational data, we *unify* the semantics of SPARQL for a wide range of annotations as follows: First, we extend mappings to annotated mappings that take values in some abstract set K of annotations. Second, we enrich K with operations for capturing the semantics of the query language operators. More specifically, we enrich K with the following three binary operations:

—A binary operator $\oplus$ for modeling $+$, $\vee$, and max, among others, for the operators π and $\cup$;
—A binary operator $\otimes$ for modeling $\times$, $\wedge$, and min, among others, for the operators $\bowtie$ and $\rtimes$; and
—A binary operator $\ominus$ for modeling $-_{bag}$, $-_{trust,}$ and $-_{fuzzy}$, among others, for the operators $\setminus$ and $\rtimes$.

Finally, based on SPARQL query equivalences that are known to hold in the bag, trust (set), and fuzzy setting, among others, we identify a set of additional properties that the algebraic structure consisting of K, $\oplus$, $\otimes$, and $\ominus$ must have, and we provide a characterization of these structures. We provide additional examples of annotated RDF commonly used in practice in Section 5 and show that these are all unified by our approach.

3. SEMANTICS OF SPARQL ON ANNOTATED RDF

In this section, we formalize the semantics of SPARQL on annotated RDF. We start by defining a general notion of annotated RDF and then extend the semantics of SPARQL correspondingly. In this section, we consider SPARQL 1.0 [Prud'hommeaux and Seaborne 2008] and discuss SPARQL 1.1 in Section 8.

We would like to emphasize that we do not consider the deductive process of obtaining implicitly entailed triples. We follow the official W3C SPARQL specification in which the semantics of SPARQL disregards the issue of RDFS reasoning. This means that SPARQL operates on the RDF graph as is, without inferring new triples. Whenever reasoning is desired, it is assumed to be carried out by a separate, underlying layer. This decision, which keeps the SPARQL query language independent from the reasoning process, brings several advantages: It results in a clean and compact semantics for SPARQL that does not interfere with reasoning rules, makes the SPARQL query language resistant to possible changes in the RDF(S) reasoning process, and allows the use of SPARQL without modifications on top of other reasoning mechanisms.

Let I, B, and L be pairwise disjoint infinite sets of Internationalized Resource Identifiers (IRIs), blank nodes, and literals, respectively. A triple $(s, p, o) \in (I \cup B) \times I \times (I \cup B \cup L)$ is called an *RDF triple*. As mentioned earlier, s is the subject, p the predicate, and o the object. An *RDF graph* is a finite set of RDF triples.

Definition 3.1. Let K be a set of annotations, disjoint from I, B, and L. A *K-annotated RDF triple* is of the form $(s, p, o) \mapsto k$ where (s, p, o) is an RDF triple and k is an

annotation taken from K. A finite set of annotated RDF triples is called a *K-annotated RDF graph* if every triple (s, p, o) has a single annotation in K.[2]

Similarly to Pérez et al. [2009], we consider a fragment of SPARQL consisting of *graph pattern expressions*, which are defined inductively as follows. Let V be a set of variables, disjoint from I, B and L.

(1) A triple from $(I \cup V) \times (I \cup V) \times (I \cup L \cup V)$ is a graph pattern, referred to as a *triple pattern*.
(2) If P_1 and P_2 are graph patterns, then $(P_1 \text{ UNION } P_2)$, $(P_1 \text{ AND } P_2)$, and $(P_1 \text{ OPT } P_2)$ are also graph patterns, referred to as a *union*, *conjunction*, and *optional graph pattern*, respectively.
(3) If P is a graph pattern and R is a SPARQL *built-in condition*, then $(P \text{ FILTER } R)$ is a graph pattern, referred to as a *filter graph pattern*.
(4) If P is a graph pattern and $S \subseteq V$, then $\text{SELECT}_S(P)$ is a graph pattern, referred to as a *projection graph pattern*.

Here, a SPARQL built-in condition is constructed using elements of the set $I \cup L \cup V$ and constants, logical connectives $(\neg, \wedge, \vee)$, inequality symbols $(<, \leqslant, \geqslant, >)$, the equality symbol $(=)$, unary predicates like bound, isBlank, and isIRI, plus other features (see Harris and Seaborne [2013] for a complete list of built-in conditions).

We next extend the semantics of SPARQL from RDF graphs to K-annotated RDF graphs, hereby closely following the presentation of the standard semantics of SPARQL given in Pérez et al. [2009]. That is, in the standard, unannotated setting, the semantics of SPARQL graph patterns on RDF graphs can be defined in terms of SPARQL algebra operations on mapping sets. Later, we first generalize mapping sets to K-annotated mapping sets, define the semantics of the SPARQL algebra operators on such mapping sets, and then extend the semantics of SPARQL in terms of the SPARQL algebra operators. The algebra operators consist of union $(\cup)$, join $(\bowtie)$, difference $(\setminus)$, left outer join $(⟕)$, projection (π), and selection (σ). We note that the difference operator is not a SPARQL operator; it was introduced by Pérez et al. [2009] to specify the semantics of the left outer join operator [Harris and Seaborne 2013].

Let $var(t)$ denote the set of variables from V that appear in a triple pattern t and denote $I \cup B \cup L$ by T. A *mapping* μ from V to T is a partial function $\mu : V \to T$. The domain of μ, denoted by $dom(\mu)$, is the subset of V on which μ is defined. We say that two mappings μ_1 and μ_2 are *compatible* if for all $v \in dom(\mu_1) \cap dom(\mu_2)$ we have $\mu_1(v) = \mu_2(v)$. We denote this by $\mu_1 \sim \mu_2$. It is readily verified that if $\mu_1 \sim \mu_2$, then $\mu_1 \cup \mu_2$ is also a mapping from $V \to T$. We denote by $\mathcal{M}$ the set of all mappings from V to T. Note that empty mappings (mappings with empty domains) are always compatible.

Definition 3.2. Let K be a set of annotations and let 0 denote a distinguished element from K. A *K-annotated mapping set on $\mathcal{M}$* is a total function $\Omega : \mathcal{M} \to K$ such that its support $\{\mu \in \mathcal{M} \mid \Omega(\mu) \neq 0\}$ is finite.

Intuitively, we interpret a mapping in $\mathcal{M}$ with its annotation set to 0 by Ω as not being part of the mapping set Ω. Definition 3.2 can thus be interpreted as saying that mapping sets Ω consist of a finite number of mappings (i.e., only a finite number of mappings have a non-zero annotation).

[2]Although the definition implies that a triple is associated with a single annotation, no generality is lost. Indeed, in application scenarios where triples may have multiple annotations from the same domain, it is common practice to combine these annotations [Zimmermann et al. 2012]. More specifically, $(s, p, o) \to k_1$ and $(s, p, o) \to k_2$ is represented by $(s, p, o) \to k_1 \oplus k_2$.

To generalize the SPARQL algebra operators to K-annotated mapping sets, we require K to be equipped with binary operations $\oplus$, $\otimes$, and $\ominus$; to contain two distinguished constants 0 and 1; and—for reasons that will be clear shortly—we assume that $\oplus$ and $\otimes$ are commutative. The definitions below are inspired by the bag, trust (set), and fuzzy semantics of SPARQL, as illustrated in Section 2. The algebra operations on mapping sets are defined as follows. Let Ω be a K-annotated mapping set on $\mathcal{M}$, $S \subseteq V$ and let R be a SPARQL built-in condition. Then, for all $\mu \in \mathcal{M}$ and $\nu \in \mathcal{M}$ such that $dom(\nu) \subseteq S$:

$$(\sigma_R(\Omega))(\mu) := \Omega(\mu) \otimes F_R(\mu)$$
$$(\pi_S(\Omega))(\nu) := \bigoplus_{\substack{\mu \in \mathcal{M}, \exists \mu' \in \mathcal{M} \\ \mu = \nu \cup \mu' \& dom(\mu') \cap S = \emptyset}} \Omega(\mu),$$

where, for all $\mu \in \mathcal{M}$, $F_R(\mu) = 1$, if μ satisfies the built-in condition R and $F_R(\mu) = 0$ otherwise. We refer to Pérez et al. [2009] for the definition of when a mapping satisfies a built-in condition. Let Ω_1 and Ω_2 be two K-annotated mapping sets. Then, for all $\mu \in \mathcal{M}$:

$$(\Omega_1 \cup \Omega_2)(\mu) := \Omega_1(\mu) \oplus \Omega_2(\mu)$$
$$(\Omega_1 \bowtie \Omega_2)(\mu) := \bigoplus_{\substack{\mu_1, \mu_2 \in \mathcal{M} \\ \mu = \mu_1 \cup \mu_2}} \Omega_1(\mu_1) \otimes \Omega_2(\mu_2)$$
$$(\Omega_1 \setminus \Omega_2)(\mu) := \Omega_1(\mu) \ominus \Big(\bigoplus_{\mu' \in \mathcal{M}, \mu \sim \mu'} \Omega_2(\mu') \Big)$$
$$(\Omega_1 \mathbin{⟕} \Omega_2)(\mu) := ((\Omega_1 \bowtie \Omega_2) \cup (\Omega_1 \setminus \Omega_2))(\mu).$$

We refer to this as the *SPARQL K-annotated algebra*. The algebra operations on K-annotated mapping sets are well-defined (i.e., they are independent of the order in which mappings in $\mathcal{M}$ are considered). Indeed, the annotations associated with the operations π_S, $\bowtie$, $\setminus$, and $\mathbin{⟕}$ involve the summation (using $\oplus$) and products (using $\otimes$) of values in K corresponding to sets of compatible mappings. The resulting annotations are independent of the order in which these compatible mappings are considered. This is a direct consequence of our assumption that $\oplus$ and $\otimes$ are commutative.

The semantics of our fragment of SPARQL on K-annotated RDF graphs is now defined inductively, following the definition of graph patterns, in terms of the SPARQL K-annotated algebra. That is, we first define the evaluation of a triple pattern t on a K-annotated RDF graph G_a as the following K-annotated mapping set. For all $\mu \in \mathcal{M}$:

$$[\![t]\!]_{G_a}(\mu) := \begin{cases} k & \text{if } dom(\mu) = var(t),\ \mu(t) \mapsto k \in G_a \\ 0 & \text{otherwise.} \end{cases}$$

Here, for a triple pattern t, we denote by $\mu(t)$ the triple obtained by replacing the variables in t according to μ. Note that this is indeed a K-annotated mapping set: (i) it defines a function because RDF triples in G_a have a single annotation, and (ii) it has finite support since RDF graphs are finite objects. The semantics of the remaining SPARQL graph patterns is defined as follows: Let P_1 and P_2 be graph patterns, R a built-in condition, and $S \subseteq V$. Then,

$$[\![P_1 \text{ FILTER } R]\!]_{G_a} := \sigma_R([\![P_1]\!]_{G_a})$$
$$[\![\text{SELECT}_S\ P_1]\!]_{G_a} := \pi_S([\![P_1]\!]_{G_a})$$
$$[\![P_1 \text{ UNION } P_2]\!]_{G_a} := [\![P_1]\!]_{G_a} \cup [\![P_2]\!]_{G_a}$$
$$[\![P_1 \text{ AND } P_2]\!]_{G_a} := [\![P_1]\!]_{G_a} \bowtie [\![P_2]\!]_{G_a}$$
$$[\![P_1 \text{ OPT } P_2]\!]_{G_a} := [\![P_1]\!]_{G_a} \mathbin{⟕} [\![P_2]\!]_{G_a}.$$

For convenience, we also consider a SPARQL graph pattern for the difference:

$$[\![P_1 \text{ DIFFERENCE } P_2]\!]_{G_a} := [\![P_1]\!]_{G_a} \setminus [\![P_2]\!]_{G_a}.$$

This graph pattern is not part of the SPARQL syntax, but, from the relationship between the SPARQL algebra operators $\ltimes$ and $\setminus$, it follows that

$$[\![P_1 \text{ OPT } P_2]\!]_{G_a} = [\![(P_1 \text{ AND } P_2) \text{ UNION } (P_1 \text{ DIFFERENCE } P_2)]\!]_{G_a}.$$

Apart from the commutativity of $\oplus$ and $\otimes$, we did not specify any other conditions on the set K and its operations. We need to impose additional conditions, however, to ensure that the SPARQL graph patterns return K-annotated mapping sets. For instance, suppose that $\oplus$ is chosen such that $0 \oplus 0 = 1$. Then, the union of two empty RDF graphs would result in a total function on $\mathcal{M}$ whose support is infinite. This contradicts the definition of K-annotated mapping sets, which demands finite support.

We identify such additional conditions on the operations in the next section. It can already be verified, however, that the preceding semantics does make sense for the following choices of $(K, \oplus, \otimes, \ominus, 0, 1)$: For $(\mathbb{N}, +, \times, -_{bag}, 0, 1)$, $(\{\mathsf{true}, \mathsf{false}\}, \vee, \wedge, -_{trust}, \mathsf{false}, \mathsf{true})$, and $([0, 1], \max, \min, -_{fuzzy}, 0, 1)$, this semantics coincides with the bag, trust (set), and fuzzy semantics of SPARQL, respectively (cf. Section 2).

4. SPARQL ANNOTATION STRUCTURE

We next identify a set of SPARQL equivalences that are expected to hold in the general K-annotated setting and show that these equivalences enforce a specific structure on the underlying set K of annotations.

4.1. SPARQL Equivalences and Identities on (K, ⊕, ⊗, ⊖, 0, 1)

Similar to the relational case [Abiteboul et al. 1995], SPARQL optimization techniques rely on SPARQL algebra *equivalences*. For the standard bag semantics of SPARQL, an extensive list of such equivalences is identified in Schmidt et al. [2010]. For example, the equivalence $\Omega_1 \cup \Omega_2 \equiv \Omega_2 \cup \Omega_1$ holds for all mapping bags Ω_1 and Ω_2. On the other hand, *identities* are commonly used to express conditions on algebraic structures such as $(K, \oplus, \otimes, \ominus, 0, 1)$. For example, the identity $x_1 \oplus x_2 = x_2 \oplus x_1$ expresses that $\oplus$ is commutative. In other words, for all values $k_1, k_2 \in K$, $k_1 \oplus k_2 = k_2 \oplus k_1$. In this case, K is said to satisfy the identity $x_1 \oplus x_2 = x_2 \oplus x_1$. Algebraic structures that satisfy a set of identities are commonly known as *equational varieties* [Burris and Sankappanavar 1981]. Equivalences and identities are always assumed to be universally quantified; we omit this quantification.

We next establish a connection between SPARQL equivalences on K-annotated RDF graphs and identities on the set K of annotations. More precisely, we show that a SPARQL K-annotated algebra equivalence $e_1 \equiv e_2$ can be translated into an identity $i_1 = i_2$ on the underlying annotation structure K.

In the following, we only consider *restricted* SPARQL K-annotated algebra equivalences because these suffice for our purpose. The left- and right-hand side of such restricted equivalences are SPARQL algebra terms built up from variables P_i, denoting arbitrary K-annotated mapping sets and $\emptyset$, denoting the empty mapping set and closed under union, join, difference, and selections of the form $\sigma_{?x=?x}$ for some variable x in V. That is, we only consider selections that originate from filter expressions in which the built-in condition vacuously holds (such as $?x = ?x$). Furthermore, restricted equivalences contain neither left-outer joins ($\ltimes$) nor projections (π) because these do not have a direct counterpart in $(K, \oplus, \otimes, \ominus, 0, 1)$.

Similarly, the left- and right-hand side of identities are terms in $(K, \oplus, \otimes, \ominus, 0, 1)$ built up from x_i, denoting arbitrary values in K, 0 and 1, two distinguished elements

Table I. Translation from SPARQL Algebra Terms to Annotation Terms by Means of the Mapping *s-to-a*

SPARQL	*s-to-a*	Annotation
P_i	$\rightarrow$	x_i
$\emptyset$	$\rightarrow$	0
$e_1 \cup e_2$	$\rightarrow$	$s\text{-}to\text{-}a(e_1) \oplus s\text{-}to\text{-}a(e_2)$
$e_1 \bowtie e_2$	$\rightarrow$	$s\text{-}to\text{-}a(e_1) \otimes s\text{-}to\text{-}a(e_2)$
$e_1 \setminus e_2$	$\rightarrow$	$s\text{-}to\text{-}a(e_1) \ominus s\text{-}to\text{-}a(e_2)$
$\sigma_{?x=?x}(e_1)$	$\rightarrow$	$s\text{-}to\text{-}a(e_1) \otimes 1$

In this table, P_i is a variable representing a graph pattern, and e_1 and e_2 are SPARQL algebra terms. Similarly, x_i is a variable representing values in K.

in K, and closed under $\oplus$, $\otimes$, $\ominus$. Table I shows the inductive definition of the mapping *s-to-a* from SPARQL terms to annotation terms.

LEMMA 4.1. *If $e_1 \equiv e_2$ is a restricted SPARQL K-annotated algebra equivalence, then $s\text{-}to\text{-}a(e_1) = s\text{-}to\text{-}a(e_2)$ is an identity that holds on $(K, \oplus, \otimes, \ominus, 0, 1)$.*

PROOF. Let $e_1 \equiv e_2$ be a SPARQL K-annotated algebra equivalence. For simplicity and without loss of generality, assume that e_1 and e_2 are SPARQL terms over the variables $P_1, \ldots, P_n$. Consider the K-annotated RDF graph G_a consisting of n triples $(a_i, b, c) \mapsto k_i$ such that $a_1, \ldots, a_n$ are n distinct constants, b and c are arbitrary constants, and $k_1, \ldots, k_n$ are arbitrary values taken from K. Furthermore, let $Q_i := \pi_{?y,?z}(\sigma_{?x=a_i}(?x, ?y, ?z))$ for $i \in [1, n]$. We denote by $e_1[[\![Q_1]\!]_{G_a}, \ldots, [\![Q_n]\!]_{G_a}]$ the SPARQL graph pattern obtained by substituting each occurrence of the variable P_i with the K-annotated mapping set $[\![Q_i]\!]_{G_a}$, for $i \in [1, n]$, and, similarly, for $e_2[[\![Q_1]\!]_{G_a}, \ldots, [\![Q_n]\!]_{G_a}]$. Along the same lines, for an annotation term t over variables $x_1, \ldots, x_n$, we denote by $t[k_1, \ldots, k_n]$ the element in K obtained by substituting occurrences of x_i by the value k_i, for $i \in [1, n]$.

Observe the following. It is readily verified that $[\![Q_i]\!]_{G_a} = \{(b, c) \mapsto k_i\}$ for $i \in [1, n]$. Furthermore, for $i, j \in [1, n]$, we have that (i) $[\![Q_i \cup Q_j]\!]_{G_a} = \{(b, c) \mapsto k_i \oplus k_j\}$; (ii) $[\![Q_i \bowtie Q_j]\!]_{G_a} = \{(b, c) \mapsto k_i \otimes k_j\}$; (iii) $[\![Q_i \setminus Q_j]\!]_{G_a} = \{(b, c) \mapsto k_i \ominus k_j\}$; and (iv) $[\![\sigma_{?w=?w}(Q_i)]\!]_{G_a} = \{(b, c) \mapsto k_i \otimes 1\}$, for $w \in \{y, z\}$. From this, and by induction on the structure of the terms e_1 and e_2, we can conclude that

$$e_i[[\![Q_1]\!]_{G_a}, \ldots, [\![Q_n]\!]_{G_a}] = \{(b, c) \mapsto s\text{-}to\text{-}a(e_i)[k_1, \ldots, k_n]\},$$

for $i = 1, 2$. Hence, $e_1 \equiv e_2$ implies that $s\text{-}to\text{-}a(e_1)[k_1, \ldots, k_n] = s\text{-}to\text{-}a(e_2)[k_1, \ldots, k_n]$ and this for arbitrary choices of $k_1, \ldots, k_n$. In other words, $s\text{-}to\text{-}a(e_1) = s\text{-}to\text{-}a(e_2)$ is an identity on K. □

The choice of annotation structure thus *entirely* depends on the SPARQL equivalences that one would like to be satisfied when evaluating SPARQL on K-annotated RDF graphs.

We next propose a set of equivalences that are desired to hold when evaluating SPARQL on K-annotated RDF graphs.

4.2. SPARQL Equivalences for the Positive Fragment

Consider first the positive fragment of the SPARQL algebra (i.e., the fragment that does not include $\setminus$ and $\bowtie$). It has been noted [Schmidt et al. 2010] that the following SPARQL algebra equivalences hold in the case of set and bag semantics and thus are natural to impose in the general K-annotated setting: For any K-annotated mapping

$$
\begin{array}{ll}
\mathsf{id}_1\colon & k\otimes 1=k\\
\mathsf{id}_2\colon & k\otimes 0=0\\
\mathsf{id}_3\colon & k\oplus 0=k\\
\mathsf{id}_4\colon & k_1\oplus k_2=k_2\oplus k_1\\
\mathsf{id}_5\colon & k_1\otimes k_2=k_2\otimes k_1\\
\mathsf{id}_6\colon & k_1\oplus(k_2\oplus k_3)=(k_1\oplus k_2)\oplus k_3\\
\mathsf{id}_7\colon & k_1\otimes(k_2\otimes k_3)=(k_1\otimes k_2)\otimes k_3\\
\mathsf{id}_8\colon & k_1\otimes(k_2\oplus k_3)=(k_1\otimes k_2)\oplus(k_1\otimes k_3)\\
 & \\
\mathsf{id}_9\colon & k\ominus k=0\\
\mathsf{id}_{10}\colon & k_1\ominus(k_2\oplus k_3)=(k_1\ominus k_2)\ominus k_3\\
\mathsf{id}_{11}\colon & k_1\otimes(k_2\ominus k_3)=(k_1\otimes k_2)\ominus k_3\\
\mathsf{id}_{12}\colon & (k_1\ominus(k_1\ominus k_2))\oplus(k_1\ominus k_2)=k_1
\end{array}
$$

Fig. 2. Axiomatization $\mathcal{A}$ of spm-semirings.

sets Ω_1, Ω_2, and Ω_3, we have that

$$
\mathcal{E}_1=\left\{\begin{array}{llll}
\sigma_{?x=?x}(\Omega_1)\equiv\Omega_1 & \Omega_1\bowtie\emptyset\equiv\emptyset & \Omega_1\cup\emptyset\equiv\Omega_1 & \Omega_1\cup\Omega_2\equiv\Omega_2\cup\Omega_1\\
\Omega_1\bowtie\Omega_2\equiv\Omega_2\bowtie\Omega_1 & \Omega_1\cup(\Omega_2\cup\Omega_3)\equiv(\Omega_1\cup\Omega_2)\cup\Omega_3 & &\\
\Omega_1\bowtie(\Omega_2\bowtie\Omega_3)\equiv(\Omega_1\bowtie\Omega_2)\bowtie\Omega_3 & \Omega_1\bowtie(\Omega_2\cup\Omega_3)\equiv(\Omega_1\bowtie\Omega_2)\cup(\Omega_1\bowtie\Omega_3) & &
\end{array}\right\}.
$$

The following proposition is a direct consequence of Lemma 4.1. Indeed, the SPARQL equivalences in $\mathcal{E}_1$ precisely correspond to the set of identities shown as id_1–id_8 in Figure 2. These identities define algebraic structures better known as semirings.

Proposition 4.2. *If the positive fragment of the SPARQL K-annotated algebra is required to satisfy the equivalences in $\mathcal{E}_1$, then $(K,\oplus,\otimes,0,1)$ must be a semiring. Furthermore, if $(K,\oplus,\otimes,0,1)$ is a semiring, then the positive SPARQL algebra operators preserve the finiteness of support. In particular, for any positive SPARQL graph pattern P and K-annotated RDF graph G_a, $[\![P]\!]_{G_a}$ has a finite support.*

Proof. Lemma 4.1 implies that the equivalences in $\mathcal{E}_1$ correspond to the semiring identities id_1–id_8 shown in Figure 2. We next show that the positive algebra operators preserve the finiteness of support. That is, if Ω, Ω_1, and Ω_2 are K-annotated mapping sets, then $\sigma_R(\Omega)$, $\pi_S(\Omega)$, $\Omega_1\cup\Omega_2$, and $\Omega_1\bowtie\Omega_2$ are K-annotated mapping sets as well (i.e., they have finite support). For $\sigma_R(\Omega)$, it is readily verified that $\sigma_R(\Omega)$ is contained in the support of Ω. Indeed, this follows from the fact that $k\otimes 1=k$ and $k\otimes 0=0$ hold in semirings. Let μ and ν be mappings in $\mathcal{M}$ such that $dom(\nu)\subseteq S\subseteq V$. Then, $(\pi_S(\Omega))(\nu)\neq 0$ iff there exists a $\mu'\in\mathcal{M}$ such that $\mu=\nu\cup\mu'$, $dom(\mu')\cap S=\emptyset$ and, furthermore, $\Omega(\mu)\neq 0$. Indeed, suppose that for all such mapping $\mu\in\mathcal{M}$, $\Omega(\mu)=0$, then $(\pi_S(\Omega))(\nu)=0\oplus\cdots\oplus 0=0$ since $0\oplus 0=0$ holds in semirings. Hence, since Ω has finite support so does $\pi_S(\Omega)$. Similarly, $(\Omega_1\cup\Omega_2)(\mu)\neq 0$ if and only if $\Omega_1(\mu)\neq 0$ or $\Omega_2(\mu)\neq 0$. Finally, $(\Omega_1\bowtie\Omega_2)(\mu)\neq 0$ if and only if there exist $\mu_1,\mu_2\in\mathcal{M}$ such that $\mu=\mu_1\cup\mu_2$ and $\Omega_1(\mu_1)\neq 0$ and $\Omega_2(\mu_2)\neq 0$. Indeed, this follows again from the fact that $k\otimes 0=0$ holds in semirings. Hence, since Ω_1 and Ω_2 have finite support so do $\Omega_1\cup\Omega_2$ and $\Omega_1\bowtie\Omega_2$. To show that $[\![P]\!]_{G_a}$ has finite support for any positive SPARQL graph pattern P and K-annotated RDF graph G_a, it suffices to observe that (i) the support of $[\![t]\!]_{G_a}$ for a triple pattern t is finite since G_a consists of finitely many triples, and (ii) $[\![P]\!]_{G_a}$ is defined inductively in terms of the positive algebra operators and triple patterns. Given that these operators preserve the finiteness of support, we may conclude that $[\![P]\!]_{G_a}$ has finite support, as desired. □

4.3. SPARQL Equivalences Involving Difference

We next turn our attention to the full SPARQL fragment including the optional operator. More specifically, in view of the relationship between $\ltimes$ and $\setminus$, we focus on the $\setminus$ operator.

The examples given in Section 2 suggest that the $\ominus$ operator corresponding to $\setminus$ is to satisfy

$$k_1 \ominus k_2 = \begin{cases} k_1 & \text{if } k_2 = 0; \text{ and} \\ 0 & \text{otherwise.} \end{cases} \qquad (\dagger)$$

Unfortunately, if we want to stay within the setting of algebraic structures defined by identities (i.e., equational varieties), then $\ominus$ cannot be defined as in (†).

PROPOSITION 4.3. *The class of algebraic structures of the form* $(K, \oplus, \otimes, \ominus, 0, 1)$ *that satisfy that for any* $k_1, k_2 \in K$, $k_1 \ominus k_2 = k_1$ *in case* $k_2 = 0$, *and* $k_1 \ominus k_2 = 0$ *otherwise, is not an equational variety.*

PROOF. This is an immediate consequence of Birkhoff's Theorem (see, e.g., Theorem 11.9 in Burris and Sankappanavar [1981]), which says that classes of algebraic structures defined by identities (i.e., equational varieties) are precisely those which are closed under taking homomorphisms, subalgebras, and products. Suppose, for the sake of contradiction, that identities exist that define the desired $\ominus$, and denote by $\mathcal{V}$ the corresponding variety. Let $(K, \oplus_K, \otimes_K, \ominus_K, 0_K, 1_K)$ and $(L, \oplus_L, \otimes_L, \ominus_L, 0_L, 1_L)$ be two algebraic structures in $\mathcal{V}$. Consider the canonical $\ominus_{K,L}$ operator on the product $K \times L$ defined as $(k_1, \ell_1) \ominus_{K,L} (k_2, \ell_2) = (k_1 \ominus_K k_2, \ell_1 \ominus_L \ell_2)$. The operations $\oplus_{K,L}$, $\otimes_{K,L}$, and constants $0_{K,L}$ and $1_{K,L}$ are defined similarly. Then, by assumption, $\mathcal{V}$ is a variety, and Birkhoff's Theorem implies that $\mathcal{V}$ must be closed under taking products. Hence, $(K \times L, \oplus_{K,L}, \otimes_{K,L}, \ominus_{K,L}, 0_{K,L}, 1_{K,L})$ must be a structure in $\mathcal{V}$ and thus $(1_K, 1_L) \ominus_{K,L} (0_K, 1_L)$ should be equal to $0_{K,L}$. However, observe that $(1_K, 1_L) \ominus_{K,L} (0_K, 1_L) = (1_K, 0_L) \neq (0_K, 0_L) = 0_{K,L}$, by the definition of $\ominus_{K,L}$. This shows that $\mathcal{V}$ is not closed under taking products, a contradiction. Hence, no set of identities exists that defines the desired $\ominus$. □

In view of Proposition 4.3, we cannot use SPARQL equivalences to precisely capture the intended semantics of $\ominus$ as given by (†). Not all is lost, however. In the following, we define an equational variety of so-called spm-semirings $(K, \oplus, \otimes, \ominus, 0, 1)$ that (i) extends the class of semirings and (ii) encompasses spm-semirings in which $\ominus$ satisfies (†). Note that Proposition 4.3 implies that there must be spm-semirings for which $\ominus$ does not satisfy (†). However, we will see later that (†) holds in most practical cases.

To define spm-semirings, we leverage Lemma 4.1. That is, we gain inspiration from known SPARQL algebra equivalences that involve $\setminus$ and then translate those into identities involving $\ominus$. More specifically, we consider the following SPARQL equivalences that involve $\setminus$ and that hold under the set and bag semantics [Schmidt et al. 2010]. For any K-annotated mapping sets Ω_1, Ω_2, and Ω_3:

$$\mathcal{E}_2 = \left\{ \begin{array}{ll} \Omega_1 \setminus \Omega_1 \equiv \emptyset & \Omega_1 \setminus (\Omega_2 \cup \Omega_3) \equiv (\Omega_1 \setminus \Omega_2) \setminus \Omega_3 \\ & \Omega_1 \bowtie (\Omega_2 \setminus \Omega_3) \equiv (\Omega_1 \bowtie \Omega_2) \setminus \Omega_3 \, (\mathit{cond}) \end{array} \right\}.$$

The last equivalence is not given in Schmidt et al. [2010] and does not hold in general. It does hold, however, under mild conditions (denoted by *cond*) on the mapping sets involved. For instance, the equivalence holds when *cond* requires that any mapping in Ω_3 that is compatible with a mapping in Ω_2 is also compatible with a mapping in Ω_1. It is easy to see that Lemma 4.1 still applies to such equivalences since the proof of that lemma uses mapping sets that satisfy this condition.

We further identify the following equivalence: For any K-annotated mapping set Ω_1 and Ω_2, we have that

$$\mathcal{E}_3 = \{(\Omega_1 \setminus (\Omega_1 \setminus \Omega_2)) \cup (\Omega_1 \setminus \Omega_2) \equiv \Omega_1\}.$$

This equivalence also holds in all settings we have considered so far. By imposing $\mathcal{E}_1$, $\mathcal{E}_2$, and $\mathcal{E}_3$ on the SPARQL K-annotated algebra, we obtain a generalization of Proposition 4.2 that accommodates for the difference ($\setminus$) and optional operator ($\mathbb{M}$). The proof again relies on Lemma 4.1.

PROPOSITION 4.4. *If the SPARQL K-annotated algebra is required to satisfy the equivalences in $\mathcal{E}_1$, $\mathcal{E}_2$, and $\mathcal{E}_3$, then $(K, \oplus, \otimes, \ominus, 0, 1)$ is an algebraic structure satisfying the identities shown in Figure 2. Furthermore, in this case, the SPARQL algebra operators preserve the finiteness of support. In particular, for any SPARQL graph pattern P and K-annotated RDF graph G_a, $[\![P]\!]_{G_a}$ has a finite support.*

PROOF. This is verified in precisely the same way as in the proof of Proposition 4.2 by using $\mathcal{E}_1$, $\mathcal{E}_2$, and $\mathcal{E}_3$ instead of $\mathcal{E}_1$ and by using id_1–id_{12} as shown in Figure 2 rather than only id_1–id_8. To show that the algebra operators preserve the finiteness of support, it suffices to show that if Ω_1 and Ω_2 are K-annotated mapping sets, then so is $\Omega_1 \mathbb{M} \Omega_2$. Indeed, we already established that the positive operators preserve the finiteness of support in Proposition 4.2, and $\Omega_1 \mathbb{M} \Omega_2$ is defined in terms of $\cup$, $\bowtie$, and $\setminus$. Let μ be a mapping in $\mathcal{M}$. Observe that $(\Omega_1 \setminus \Omega_2)(\mu)$ can be different from 0 only if $\Omega_1(\mu) \neq 0$. Indeed, this follows from the fact that $0 \ominus k = 0$ holds in spm-semirings:

$$\begin{aligned} 0 \ominus k &= (0 \otimes 1) \ominus k && \text{(by } \mathsf{id}_1\text{)} \\ &= 0 \otimes (1 \ominus k) && \text{(by } \mathsf{id}_{11}\text{)} \\ &= 0. && \text{(by } \mathsf{id}_2\text{)} \end{aligned}$$

Finally, in precisely the same way as in Proposition 4.2, one can verify that $[\![P]\!]_{G_a}$ has a finite support for any SPARQL expression P and K-annotated RDF graph G_a. □

Definition 4.5. An *spm-semiring* is an algebraic structure $(K, \oplus, \otimes, \ominus, 0, 1)$ that satisfies the identities id_1–id_{12} given in Figure 2. Here, "spm" stands for "SPARQL minus." The class of spm-semirings thus forms an equational variety.

Proposition 4.4 thus implies that spm-semirings are a good candidate annotation structure when considering SPARQL on annotated RDF. It can be readily verified that $(\mathbb{N}, +, \times, -_{bag}, 0, 1)$, $(\{\mathsf{true}, \mathsf{false}\}, \vee, \wedge, -_{trust}, \mathsf{false}, \mathsf{true})$, and $([0, 1], \max, \min, -_{fuzzy}, 0, 1)$ are spm-semirings. We will see more examples of spm-semirings in the next section.

4.4. Minimality

We next address the minimality of the set of identities that define spm-semirings. Let $\mathcal{I}$ be a set of identities and K an algebraic structure. We say that K *satisfies* $\mathcal{I}$, denoted by $K \models \mathcal{I}$, if K satisfies all identities in $\mathcal{I}$. Let e be an identity. Then e is *implied* by $\mathcal{I}$, denoted by $\mathcal{I} \models e$, if for all structures K such that $K \models \mathcal{I}$ we have $K \models e$. Two sets of identities $\mathcal{I}_1$ and $\mathcal{I}_2$ are said to be equivalent if all identities in $\mathcal{I}_2$ are implied by $\mathcal{I}_1$ and vice versa. A set $\mathcal{I}$ of identities is said to be *minimal* if for all $e \in \mathcal{I}$, $(\mathcal{I} \setminus e) \not\models e$. Let $\mathcal{A}$ be the set of spm-semiring identities shown in Figure 2.

PROPOSITION 4.6. *The sets of identities $\mathcal{A}$ and $\mathcal{A} \setminus \{\mathsf{id}_2, \mathsf{id}_3\}$ are equivalent and, furthermore, $\mathcal{A} \setminus \{\mathsf{id}_2, \mathsf{id}_3\}$ is minimal. In other words, $\mathcal{A} \setminus \{\mathsf{id}_2, \mathsf{id}_3\}$ is a minimal set of identities defining the class of spm-semirings.*

PROOF. Let $\mathcal{A}' = \mathcal{A} \setminus \{\mathsf{id}_2, \mathsf{id}_3\}$. We first show minimality. It suffices to verify that for each identity id in $\mathcal{A}'$ there exists a structure that satisfies all identities in $\mathcal{A}' \setminus \{\mathsf{id}\}$ but violates id. The list of these structures is deferred to the appendix. For the equivalence of $\mathcal{A}$ and $\mathcal{A}'$, it suffices to verify $\mathcal{A}' \models \{\mathsf{id}_2, \mathsf{id}_3\}$ since $\mathcal{A}' \subset \mathcal{A}$. We first show that $\mathcal{A}'$ implies that for any $k_1 \in K$, $0 \oplus (k_1 \ominus 0) = k_1$.

$$\begin{aligned}
k_1 &= \big(k_1 \ominus (k_1 \ominus k_2)\big) \oplus (k_1 \ominus k_2) && (\text{by } \mathsf{id}_{12})\\
&= \big(k_1 \ominus (k_1 \ominus k_1)\big) \oplus (k_1 \ominus k_1) && (\text{for } k_2 = k_1)\\
&= (k_1 \ominus 0) \oplus 0 && (\text{by } \mathsf{id}_9)\\
&= 0 \oplus (k_1 \ominus 0). && (\text{by } \mathsf{id}_4)
\end{aligned}$$

In particular, $0 = 0 \oplus (0 \ominus 0)$, which in turn is equal to $0 \oplus 0$ by id_9. From this, it follows that $\mathcal{A}' \models \mathsf{id}_3$:

$$\begin{aligned}
k_1 &= 0 \oplus (k_1 \ominus 0) && (\text{by the above identity})\\
&= (0 \oplus 0) \oplus (k_1 \ominus 0) && (\text{by } 0 = 0 \oplus 0)\\
&= 0 \oplus (0 \oplus (k_1 \ominus 0)) && (\text{by } \mathsf{id}_6)\\
&= 0 \oplus k_1 && (\text{by the above identity})\\
&= k_1 \oplus 0. && (\text{by } \mathsf{id}_4)
\end{aligned}$$

We can thus safely use id_3 to show that $\mathcal{A}' \models \mathsf{id}_2$:

$$\begin{aligned}
k_1 \otimes 0 &= k_1 \otimes (k_1 \ominus k_1) && (\text{by } \mathsf{id}_9)\\
&= k_1 \otimes (k_1 \ominus (k_1 \otimes 1)) && (\text{by } \mathsf{id}_1)\\
&= k_1 \otimes (k_1 \ominus (k_1 \otimes (1 \oplus 0))) && (\text{by } \mathsf{id}_2)\\
&= k_1 \otimes (k_1 \ominus ((k_1 \otimes 1) \oplus (k_1 \otimes 0))) && (\text{by } \mathsf{id}_8)\\
&= k_1 \otimes (k_1 \ominus (k_1 \oplus (k_1 \otimes 0))) && (\text{by } \mathsf{id}_1)\\
&= k_1 \otimes ((k_1 \ominus k_1) \ominus (k_1 \otimes 0)) && (\text{by } \mathsf{id}_{10})\\
&= k_1 \otimes (0 \ominus (k_1 \otimes 0)) && (\text{by } \mathsf{id}_9)\\
&= (k_1 \otimes 0) \ominus (k_1 \otimes 0) && (\text{by } \mathsf{id}_{11})\\
&= 0. && (\text{by } \mathsf{id}_9)
\end{aligned}$$

In other words, $\mathcal{A}' \models \{\mathsf{id}_2, \mathsf{id}_3\}$. □

5. CHARACTERIZATION OF SPM-SEMIRINGS

With the definition of spm-semirings at hand, an obvious question is how to construct spm-semirings from algebraic objects that we already know. In this section, we characterize the class of spm-semirings in terms of a combination of semirings and boolean algebras, two standard algebraic structures (Theorem 5.3). Not only does this characterization allow us to show that certain structures are spm-semirings, it also opens the way for identifying a universal object in the class of spm-semirings, as will be shown in the next section.

5.1. Seba-Structures: Semirings with an Embedded Boolean Algebra

Recall from Section 4 that we would have preferred $\ominus$ to satisfy (†); that is, for any $k_1, k_2 \in K$, $k_1 \ominus k_2 = 0$ if $k_2 \neq 0$, and $k_1 \ominus k_2 = k_1$ otherwise. Or, in other words, the value of $k_1 \ominus k_2$ is determined by k_1 and whether or not k_2 is equal to 0.

To get some inspiration on how to obtain an operator $\ominus$ that satisfies (†) by combining semirings with boolean algebras, we consider a semiring K and the two-element boolean

algebra $B_2 = (\{\bot, \top\}, \vee, \wedge, \bar{}, \bot, \top)$. Observe the following: Assume that we have some way of mapping elements k in K to elements in B_2 such that k is mapped to $\top$ if k is different from 0 and to $\bot$ if k is equal to 0. Let us denote such mapping from K to B_2 by d. Furthermore, assume that we can embed the boolean values $\top$ and $\bot$ to 1 and 0 in K, respectively. Denote such an embedding from B_2 to K by $\imath$. Define $k_1 \ominus k_2 := k_1 \otimes \imath(\overline{d(k_2)})$, where $\overline{d(k_2)}$ is the complement of $d(k_2)$. Then clearly, $k_1 \ominus k_2$ satisfies (†).

We next identify some properties that these mappings d and $\imath$ should have. Clearly, $d(1) = \top$ and $d(0) = \bot$ must hold for d. Similarly, the mapping $\imath$ must satisfy $\imath(\bot) = 0$ and $\imath(\top) = 1$. These conditions pose some additional restrictions on how the operations in K interact with the operations in B_2. Indeed, one would be tempted to require that for elements $k_1, k_2 \in K$ and $b_1, b_2 \in B_2$ we have that $d(k_1 \oplus k_2) = d(k_1) \vee d(k_2)$, $d(k_1 \otimes k_2) = d(k_1) \wedge d(k_2)$, $\imath(b_1 \vee b_2) = \imath(b_1) \oplus \imath(b_2)$, and $\imath(b_1 \wedge b_2) = \imath(b_1) \otimes \imath(b_2)$; that is, d and $\imath$ are proper embeddings of K in B_2 and vice versa, respectively. Some of these requirements, however, are too restrictive and limit the kind of semirings that we can consider.

For example, $\imath(b_1 \vee b_2) = \imath(b_1) \oplus \imath(b_2)$ implies that $1 = \imath(\top) = \imath(\top \vee \top) = \imath(\top) \oplus \imath(\top) = 1 \oplus 1$, which is a property that does not hold for general semirings K. Instead, we want that $\imath(\top \vee \top) = \imath(\top)$, $\imath(\top \vee \bot) = \imath(\top)$ and $\imath(\bot \vee \bot) = \imath(\bot)$. This can be achieved, for instance, by defining $\imath(b_1 \vee b_2) = \imath(b_1) \oplus (\imath(\overline{b_1}) \otimes \imath(b_2))$ for elements $b_1, b_2 \in B_2$. Indeed, observe that $\imath(\top \vee \top) = \imath(\top) \oplus (\imath(\bot) \otimes \imath(\top)) = \imath(\top)$, as desired.

As another example, consider a semiring K with zero divisors; that is, there exist $k_1, k_2 \in K$ such that $k_1 \neq 0$, $k_2 \neq 0$ but $k_1 \otimes k_2 = 0$. Then, $\bot = d(k_1 \otimes k_2) = d(k_1) \wedge d(k_2) = \top \wedge \top = \top$ does not hold. Nevertheless, we know that $\bot = d(k \otimes 0) = d(k) \wedge \bot$ and, similarly, $d(k) = d(k \otimes 1) = d(k) \wedge \top$ for any $k \in K$. We therefore only require that $d(k \otimes \imath(b)) = d(k) \wedge b$ for $k \in K$ and $b \in B_2$.

As we show later, $d(k_1 \oplus k_2) = d(k_1) \vee d(k_2)$, $d(k \otimes \imath(b)) = d(k) \wedge b$, $\imath(b_1 \vee b_2) = \imath(b_1) \oplus (\imath(\overline{b_1}) \otimes \imath(b_2))$, and $\imath(b_1 \wedge b_2) = \imath(b_1) \otimes \imath(b_2)$ allow us to define $k_1 \ominus k_2 := k_1 \otimes \imath(\overline{d(k_2)})$ in general, provided that we add one more condition involving complementation in B_2. In particular, since we want that $k \ominus k = 0$, we assume that $k \otimes \imath(\overline{d(k)}) = 0$ holds for d and $\imath$.

In the following definitions, we generalize these observations by means of a so-called *seba*-structure that consists of a semiring K together with an embedded boolean algebra B, linked together with mappings $d : K \to B$ and $\imath : B \to K$.

Definition 5.1. A *seba-structure* is of the form $(K, B, d, \imath)$ where K is a commutative semiring $(K, \oplus, \otimes, 0, 1)$, B a boolean algebra $(B, \vee, \wedge, \bar{}, \bot, \top)$, and $d : K \to B$ and $\imath : B \to K$ are mappings such that

sb_1:	$d(0) = \bot$ and $d(1) = \top$	sb_2:	$\imath(\bot) = 0$ and $\imath(\top) = 1$
sb_3:	$d(k_1 \oplus k_2) = d(k_1) \vee d(k_2)$	sb_4:	$\imath(b_1 \vee b_2) = \imath(b_1) \oplus \big(\imath(\bar{b}_1) \otimes \imath(b_2)\big)$
sb_5 :	$d(k \otimes \imath(b)) = d(k) \wedge b$	sb_6:	$\imath(b_1 \wedge b_2) = \imath(b_1) \otimes \imath(b_2)$
sb_7:	$k \otimes \imath(\overline{d(k)}) = 0$. □		

Given this, we next show how $\ominus$ can be defined in terms of a seba-structure following our earlier observation.

Definition 5.2. We say that an algebraic structure $(L, \oplus, \otimes, \ominus, 0, 1)$ is *derived* from a seba-structure $(K, B, d, \imath)$ if $(L, \oplus, \otimes, 0, 1)$ and $(K, \oplus, \otimes, 0, 1)$ coincide and for any $k, \ell \in L$, $k \ominus \ell = k \otimes \imath(\overline{d(\ell)})$.

The main result of this section is that spm-semirings and seba-structures are closely related.

Table II. Spm-Semirings and Their Applications

Spm-semiring	Application
$(\{\mathsf{true}, \mathsf{false}\}, \vee, \wedge, -_{trust}, \mathsf{false}, \mathsf{true})$	Trust/Set Semantics
$(N^{\infty}, \min, +, -_{rtrust}, \infty, 0)$	Ranked Trust
$(\mathcal{C}, \min, \max, -_{acc}, 0, P)$	Access control
$(\mathcal{P}(E), \cup, \cap, -_{prob}, \emptyset, E)$	Event Tables
$(\mathbb{N}, +, \times, -_{bag}, 0, 1)$	Bags
$([0, 1], \max, \min, -_{fuzzy}, 0, 1)$	Fuzzy

THEOREM 5.3. *Every spm-semiring is derived from some seba-structure, and, vice versa, every structure derived from a seba-structure is an spm-semiring.*

The proof of this theorem is deferred to the appendix.

5.2. Seba-Structures and Derived Spm-Semirings: Examples

We next provide some examples of seba-structures and their derived spm-semirings. Table II summarizes some of the spm-semirings and their application domains.

Example 5.4 (*Trust, Set*). Consider the relational trust semiring $T = (\{\mathsf{true}, \mathsf{false}\}, \vee, \wedge, \mathsf{false}, \mathsf{true})$ [Green et al. 2007]. The semiring T is readily extended to a boolean algebra T_b by incorporating complementation $\overline{\ }$. Consider T_b together with the mappings $d : T \to T_b$ and $\iota : T_b \to T$, both of which are the identity mappings. It is readily verified that (T, T_b, d, ι) is a seba-structure. The derived spm-semiring consists of T together with the additional operator $b_1 \ominus b_2 := b_1 \wedge \overline{b_2}$. Note that $\ominus$ coincides with $-_{trust}$ (cf. Example 2.3).

Example 5.5 (*Boolean Algebras*). The previous example generalizes to any semiring K that can be extended to a boolean algebra K_b. As in the previous example, (K, K_b, d, ι) with d and ι, the identity mapping then forms a seba-structure. The derived spm-semiring in this case is given by K extended with $k_1 \ominus k_2 := k_1 \wedge \overline{k_2}$. In other words, $\ominus$ coincides with the standard notion of difference in boolean algebras. For example, consider the relational probability semiring [Green et al. 2007] $(\mathcal{P}(E), \cup, \cap, \emptyset, E)$ for a set E of events. Clearly, this semiring can be equipped with complementation and is thus part of a boolean algebra. As a consequence, the $\ominus$ operator in the derived spm-semiring is given by $E_1 \cap E_2^c$, where E^c denotes the complement of E.

The previous example shows that the $\ominus$ operator in spm-semirings does not always satisfy $k_1 \ominus k_2 = k_1$ in case $k_2 = 0$ and $k_1 \ominus k_2 = 0$ otherwise. Indeed, consider the two subsets $A = \{a, b\}$ and $B = \{c, d\}$ of $X = \{a, b, c, d\}$. Then, $A \cap B^c = A$ despite the fact that $B \neq \emptyset$. Here, complementation is taken relative to X. In view of Proposition 4.3, this is not unexpected, however. We will see later that the probability semiring can be extended to an spm-semiring in another way such that the operator $\ominus$ behaves as intended.

Example 5.6 (*Bags, Fuzzy*). Let $B_2 = (\{0, 1\}, \vee, \wedge, \overline{\ }, 0, 1)$ be the two-element boolean algebra and let $(\mathbb{N}, +, \times, 0, 1)$ be the semiring of natural numbers. Define $d : \mathbb{N} \to \{0, 1\}$ as $d(x) = 0$ if $x = 0$ and $d(x) = 1$ if $x \neq 0$. Let $\iota : \{0, 1\} \to \mathbb{N}$ be the identity mapping. Then, clearly, $d(0) = 0, d(1) = 1, \iota(0) = 0$ and $\iota(1) = 1$. Furthermore, it is readily verified that for any $x, y \in \mathbb{N}$ and $b, b_1, b_2 \in \{0, 1\}$, $d(x + y) = d(x) \vee d(y)$, $d(x \times \iota(b)) = d(x) \wedge b$, $\iota(b_1 \vee b_2) = \iota(b_1) + (\iota(\overline{b_1}) \times \iota(b_2)) = b_1 + (\overline{b_1} \times b_2)$, and $\iota(b_1 \wedge b_2) = \iota(b_1) \times \iota(b_2) = b_1 \times b_2$. We also have that $x \times \iota(\overline{d(x)}) = 0$. Hence, $(\mathbb{N}, B_2, d, \iota)$ is a seba-structure. We can thus extend $\mathbb{N}$ with $\ominus$ derived from $(\mathbb{N}, B_2, d, \iota)$ by letting $x \ominus y = x \times \overline{d(y)}$, for any $x, y \in \mathbb{N}$. That is, $x \ominus y = 0$ if $y \neq 0$ and $x \ominus y = x$ otherwise. Note that $\ominus$ coincides with $-_{bag}$ when SPARQL is evaluated on RDF under—the default—bag semantics (cf. Example 2.1).

Along the same lines, the fuzzy semiring $F = ([0, 1], \max, \min, 0, 1)$ can be extended with $\ominus$ derived from (F, B_2, d, ι) with d and ι, as above. Note that, in this case, $p \ominus q = \min\{p, \overline{d(q)}\}$ is equal to 0 if $q \neq 0$, and is p otherwise; that is, $\ominus$ coincides with $-_{fuzzy}$, as desired (cf. Example 2.4).

Example 5.7 (*Zero-Sum Free Semirings*). The previous example can be generalized to arbitrary zero-sum free semirings. Recall that a semiring K is zero-sum free if for any $k, \ell \in K$ we have that $k \oplus \ell = 0$ implies that both k and ℓ are 0. It is readily verified that for such K, (K, B_2, d, ι) is a seba-structure, where $d(k) = 0$ if $k = 0$ and $d(k) = 1$ if $k \neq 0$, and $\iota(0) = 0_K$ and $\iota(1) = 1_K$. The $\ominus$ operator in the derived structure is consequently defined as $k \ominus \ell := k \otimes \iota(\overline{d(\ell)})$.

The restriction to zero-sum free semirings in the previous example is necessary. Indeed, let (K, B, d, ι) be a seba-structure and assume that $k \oplus \ell = 0$. Then also $d(k \oplus \ell) = \bot = d(k) \vee d(\ell)$ and hence both $d(k) = \bot$ and $d(\ell) = \bot$. We claim that $d(k) = \bot$ if and only if $k = 0$. Suppose, for the sake of contradiction, that $d(k) = \bot$ for $k \neq 0$. Then, from $0 \neq k = k \otimes \iota(\overline{d(k)}) = 0$ we obtain a contradiction. Similarly, for $d(\ell) = \bot$.

Example 5.8 (*Probability*). Let us reconsider the probability semiring $(\mathcal{P}(E), \cup, \cap, \emptyset, E)$. Clearly, this semiring is zero-sum free, and the previous example tells us that a seba-structure can be obtained as follows: For any event $E_i \subseteq E$, $d(E_i) = 1$ if $E_i \neq \emptyset$ and $d(E_i) = 0$ otherwise, and $\iota(0) = \emptyset$ and $\iota(1) = E$. In other words, we map events in $\mathcal{P}(E)$ to the two-element boolean algebra B_2 such that the empty event is mapped to 0 and all non-empty events are mapped to 1. As a consequence, the operator $E_i \ominus E_j$ on $\mathcal{P}(E)$ in the derived spm-semiring is defined by $E_i \cap \iota(\overline{d(E_j)})$ and results in E_i when E_j is empty and in $\emptyset$ otherwise. To relate the semantics of $E_i \ominus E_j$ with the standard possible world semantics of probabilistic RDF and the certain answer interpretation of SPARQL (see, e.g., Kementsietsidis et al. [2014]) observe the following. If the probability $p(E_j)$ of E_j is zero, then there is no possible world in which E_j appears, and thus E_i should be returned in every possible world. The certain answers thus would be E_i, consistent with $E_i \ominus E_j$. In contrast, if $p(E_j) > 0$, then there is at least one world in which E_j is returned, and, on this world, an empty result should be returned. As a consequence, the certain answer would be empty as well, consistent with $E_i \ominus E_j$. We also denote $\ominus$ by $-_{prob}$.

Example 5.9 (*Ranked Trust, Access Policy*). Consider the tropical semiring $(\mathbb{N}^\infty, \min, +, \infty, 0)$ [Green et al. 2007], where $\mathbb{N}^\infty$ is short for $\mathbb{N} \cup \{\infty\}$. This semiring has been used to model ranked trust for relational queries [Karvounarakis et al. 2010] and can be regarded as a generalization of the boolean trust semiring. In the case of RDF data, triples that are completely untrusted and should be disregarded have ∞ as their rank, whereas completely trusted triples have rank 0. Clearly, this is a zero-sum-free semiring, and, similarly to the previous examples, one can extend it to an spm-semiring such that $m -_{rtrust} n = \infty$ in case $n \neq \infty$, and $m -_{rtrust} n = m$ otherwise. Along the same lines, one can extend the semiring $(\mathcal{C}, \min, \max, 0, P)$ [Foster et al. 2008], with $\mathcal{C} = \{P(\text{ublic}), C(\text{onfidential}), S(\text{ecret}), T(\text{op Secret}), 0\}$, used in the context of confidentiality policies, to an spm-semiring. Here, the order between the levels of access is specified as $P < C < S < T < 0$, and the operators min and max are defined relative to this order. It is readily verified that, in the resulting spm-semiring, $v -_{acc} w = v$ if w is 0 and can thus be ignored, and $v -_{acc} w = 0$ otherwise.

6. UNIVERSAL SPM-SEMIRING

The importance of the universal "most general" object in a class of algebraic structures has already been emphasized in the relational context [Green et al. 2007]. Indeed, in

that setting, semirings are the appropriate annotation structure and the semiring of polynomials $(\mathbb{N}[X], +, \cdot, 0, 1)$ is known to be universal. It has been shown that the evaluation of positive relational algebra expressions on semiring-annotated relations *factors through* the evaluation on relations that take their annotations from $(\mathbb{N}[X], +, \cdot, 0, 1)$ [Green et al. 2007]. This implies, among other things, that it suffices to extend positive relational algebra evaluation algorithms to relations that are annotated with values in the universal object from which the query results on specific annotation structures (semirings) can then easily be obtained. In this section, we first identify a universal object in the class of seba-structures for which we then show that the derived structure is a universal spm-semiring. In Section 7, we establish a factorization property for SPARQL evaluation by leveraging this universal spm-semiring.

6.1. Universal Seba-Structure

We first define the notion of universal seba-structure. Let $(K_1, \oplus_1, \otimes_1, 0_1, 1_1)$ and $(K_2, \oplus_2, \otimes_2, 0_2, 1_2)$ be two semirings; $(B_1, \vee_1, \wedge_1, {}^{-1}, \bot_1, \top_1)$ and $(B_2, \vee_2, \wedge_2, {}^{-2}, \bot_2, \top_2)$ be two boolean algebras; and $d_1 : K_1 \to B_1$, $d_2 : K_2 \to B_2$, $\iota_1 : B_1 \to K_1$ and $\iota_2 : B_2 \to K_2$ be mappings such that (K_1, B_1, d_1, ι_1) and (K_2, B_2, d_2, ι_2) are seba-structures. We say that a mapping $(h, \beta) : (K_1, B_1) \to (K_2, B_2)$ is a *seba-homomorphism* if the following conditions are satisfied:

—The mapping h is a semiring homomorphism from K_1 to K_2. That is, for any $k, k' \in K_1$, $h(k \oplus_1 k') = h(k) \oplus_2 h(k')$, and $h(k \otimes_1 k') = h(k) \otimes_2 h(k')$. Furthermore, $h(0_1) = 0_2$ and $h(1_1) = 1_2$.
—The mapping β is a boolean algebra homomorphism from B_1 to B_2. That is, for any $b, b' \in B_1$, $\beta(b \vee_1 b') = \beta(b) \vee_2 \beta(b')$, $\beta(b \wedge_1 b') = \beta(b) \wedge_2 \beta(b')$, $\overline{\beta(b)}^2 = \beta(\overline{b}^1)$, $\beta(\bot_1) = \bot_2$, $\beta(\top_1) = \top_2$; and finally,
—$\beta(d_1(k)) = d_2(h(k))$ and $h(\iota_1(b)) = \iota_2(\beta(b))$.

Let $X = \{x_1, \ldots, x_n\}$ be a finite set of variables. A semiring $(K, \oplus, \otimes, 0, 1)$ is said to be finitely generated by X if it is the smallest semiring such that $X \subseteq K$. A seba-structure (K, B, d, ι) is finitely generated by X if its semiring is finitely generated by X. We say that (K, B, d, ι) is freely generated by X if it is generated by X, and, in addition, for every seba-structure (K_1, B_1, d_1, ι_1) and any valuation $\nu : X \to K_1$, we can uniquely extend ν to a seba-homomorphism (h, β) from (K, B, d, ι) to (K_1, B_1, d_1, ι_1) such that h coincides with ν on X. In this case, we call (K, B, d, ι) a universal object in the class of seba-structures relative to the generator set X. It can be shown that such a universal object is unique up to isomorphism (using seba-isomorphisms). We refer to Burris and Sankappanavar [1981] for the general theory of universal objects.

A few comments are in order here:
(1) It may seem surprising that no explicit set of generators for the boolean algebra B is provided. The reason is that the generators for B are determined by X as well. To see this, observe that in a seba-structure (K, B, d, ι), it is always the case that $B = d(K)$. Indeed, for any element $b \in B$, $\iota(b) \in K$ and $d(\iota(b)) = d(1 \otimes \iota(b)) = d(1) \wedge b = \top \wedge b = b$. If K is finitely generated by X, then we have that $X \subseteq K$ and, consequently, K contains all terms built up from elements in X, 0, 1 and using the semiring operations $\oplus$ and $\otimes$. Clearly, B must contain $d(0)$ and $d(1)$. Furthermore, sb_3 tells us that d is compatible with $\oplus$, and thus elements in B are determined by $d(\mu)$, where μ belongs to the set $mon(X)$ of monomials over X. For example, if $X = \{x_1, x_2\}$ then $mon(X) = \{x_1^{\otimes m} \otimes x_2^{\otimes n} \mid m, n \geqslant 0\}$ where $x_1^{\otimes m} = \underbrace{x_1 \otimes \cdots \otimes x_1}_{m\text{-times}}$, similarly for $x_2^{\otimes n}$. For ease of notation, we often use the empty notation instead of $\otimes$ and simply write x_1^m, x_2^n, and $x_1^m x_2^n$ instead of $x_1^{\otimes m}$, $x_2^{\otimes n}$ and $x_1^{\otimes m} \otimes x_2^{\otimes n}$, respectively. For instance, for $X = \{x_1, x_2\}$, we write $mon(X) = \{1 = x_1^0 x_2^0,$

$x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, \ldots\}$. We thus have that B can be regarded as the smallest boolean algebra that contains $d(0)$, $d(1)$, and $d(\mu)$ for $\mu \in mon(X)$. The generators of B are thus implicitly determined by 0, 1, X, and the mapping $d : K \to B$.
(2) For a mapping $(h, \beta) : (K_1, B_1) \to (K_2, B_2)$ to be a seba-homomorphism, it suffices that h is a semiring homomorphism, β is compatible with complementation, and $\beta(d_1(k)) = d_2(h(k))$ and $h(\iota_1(b)) = \iota_2(\beta(b))$ hold. To see this, observe that $\beta(b \vee_1 b') = \beta(\overline{\overline{b}^1 \wedge_1 \overline{b'}^1}^1) = \overline{\beta(\overline{b}^1 \wedge_1 \overline{b'}^1)}^2$. Furthermore, $\beta(\overline{b}^1 \wedge_1 \overline{b'}^1) = \beta(d_1(\iota_1(\overline{b}^1) \otimes_1 \iota_1(\overline{b'}^1))) = d_2(h(\iota_1(\overline{b}^1)) \otimes_2 h(\iota_2(\overline{b'}^1)))$, which in turn is equal to $d_2(\iota_2(\overline{\beta(b)}^2) \otimes_2 \iota_2(\overline{\beta(b')}^2)) = \overline{\beta(b)}^2 \wedge_2 \overline{\beta(b')}^2$. Hence, $\beta(b \vee_1 b') = \beta(b) \vee_2 \beta(b')$. In a similar way, one can verify that $\beta(b \wedge_1 b') = \beta(b) \wedge_2 \beta(b')$. Finally, $\beta(\top_1) = \beta(d_1(1_1)) = d_2(h(1_1)) = d_2(1_2) = \top_2$ and similarly, $\beta(\bot_1) = \bot_2$.

In the following, we construct a universal seba-structure by extending polynomials with boolean variables. More specifically, we first define a semiring of so-called boolynomials. Second, we show that this semiring has a boolean algebra embedded in it. The elements of this algebra correspond to polynomials consisting of boolean variables only. Finally, we define the mappings d and ι between the semiring and boolean algebra and show that, all combined, these form a seba-structure.

6.2. Semiring of Boolynomials

Let $X = \{x_1, \ldots, x_n\}$ be a set of variables and $mon(X)$ be the set of monomials over X. We define two other sets of variables, denoted by B_X and $\bar{B}_X$, that are disjoint from X, as follows: $B_X = \{b_\mu \mid \mu \in mon(X)\}$ and $\bar{B}_X = \{\bar{b}_\mu \mid \mu \in mon(X)\}$. To simplify notation, we denote the set of variables $X \cup B_X \cup \bar{B}_X$ by X_B.

Let $\mathbb{N}[X_B]$ be the set of polynomials with coefficients in $\mathbb{N}$ over the variables X_B. The semiring of boolynomials is then defined as follows:

First, we define a congruence relation θ on the semiring of polynomials $(\mathbb{N}[X_B], +, \cdot, 0, 1)$ indicating when two polynomials are considered to be equivalent. Recall that θ is a congruence relation on $\mathbb{N}[X_B]$ if it is an equivalence relation satisfying the additional conditions that if $(p[X_B], q[X_B]) \in \theta$ and $(p'[X_B], q'[X_B]) \in \theta$, then $(p[X_B] + p'[X_B], q[X_B] + q'[X_B]) \in \theta$ and $(p[X_B] \cdot p'[X_B], q[X_B] \cdot q'[X_B]) \in \theta$.

Second, we consider the quotient semiring of $\mathbb{N}[X_B]$ with respect to θ. This semiring has as elements the equivalence classes $[p[X_B]]$ consisting of all polynomials that are congruent to $p[X_B]$, operations $[p[X_B]] + [q[X_B]] = [p[X_B] + q[X_B]]$, $[p[X_B]] \cdot [q[X_B]] = [p[X_B] \cdot q[X_B]]$, and constants [0] and [1].

Definition 6.1. The *semiring of boolynomials* is the quotient semiring of $(\mathbb{N}[X_B], +, \cdot, 0, 1)$ with respect to θ. We denote this semiring by $(\mathbb{N}_\theta[X_B], +, \cdot, 0, 1)$[3].

It remains to define the congruence relation θ. Intuitively, this relation is to capture that the variables in B_X and $\bar{B}_X$ represent booleans such that b_μ indicates that the monomial μ is considered being evaluated to a non-zero element, whereas $\bar{b}_\mu$ indicates the opposite. We define θ as the smallest congruence relation on $\mathbb{N}[X_B]$ such that it contains the following pairs (equivalences). For all monomials $\mu \in mon(X)$:

(i) $(b_\mu \cdot \bar{b}_\mu, 0) \in \theta$ and $(b_\mu + \bar{b}_\mu, 1) \in \theta$, indicating that b_μ and $\bar{b}_\mu$ are complements.
(ii) $(b_\mu + b_\mu, b_\mu) \in \theta$ and $(\bar{b}_\mu + \bar{b}_\mu, \bar{b}_\mu) \in \theta$, to indicate that "+" is idempotent for variables in B_X and $\bar{B}_X$.
(iii) $(\bar{b}_\mu \cdot \mu, 0) \in \theta$ indicating that the presence of $\bar{b}_\mu$ implies that μ should be evaluated to zero.

[3]We abuse notation and use $+$, $\cdot$, 0, and 1 for the operations and constants in $\mathbb{N}_\theta[X_B]$, respectively.

(iv) $(b_\mu \cdot b_\nu, b_\mu) \in \theta$ whenever $\mu = \nu \cdot \nu'$ for some monomial ν', indicating that whenever μ evaluates to non-zero, then any more general monomial ν evaluates to non-zero as well. Here, ν is more general than μ if $\mu = \nu \cdot \nu'$ for some monomial ν'.

Note that (i) and (ii) imply that $[b_\mu] = [b_\mu] \cdot ([b_\mu] + [\bar{b}_\mu]) = [b_\mu] \cdot [b_\mu]$ and $[\bar{b}_\mu] = [\bar{b}_\mu] \cdot ([b_\mu] + [\bar{b}_\mu]) = [\bar{b}_\mu] \cdot [\bar{b}_\mu]$. In other words, also "$\cdot$" is idempotent for variables in B_X and $\bar{B}_X$. Furthermore, (i) and (iii) imply that $[\mu] = ([b_\mu] + [\bar{b}_\mu]) \cdot [\mu] = [b_\mu] \cdot [\mu] + [\bar{b}_\mu] \cdot [\mu] = [b_\mu] \cdot [\mu]$. That is, the presence of μ implies the presence of b_μ. In addition, from (i) and (iv) it follows that $[\bar{b}_\nu] = ([b_\mu] + [\bar{b}_\mu]) \cdot [\bar{b}_\nu] = [b_\mu] \cdot [\bar{b}_\nu] + [\bar{b}_\mu] \cdot [\bar{b}_\nu] = [b_\mu] \cdot [b_\nu] \cdot [\bar{b}_\nu] + [\bar{b}_\mu] \cdot [\bar{b}_\nu] = [\bar{b}_\mu] \cdot [\bar{b}_\nu]$ whenever $\mu = \nu \cdot \nu'$. That is, whenever ν evaluates to zero then so does every more specific monomial μ. Here, μ is more specific than ν if $\mu = \nu \cdot \nu'$ for some monomial ν'. Also observe that $[1] = [b_1]$ and $[0] = [\bar{b}_1]$ where b_1 corresponds to the constant monomial $1 = x_1^0 \otimes \cdots \otimes x_n^0$.

To gain some insight in the semiring of boolynomials, we next provide a procedure for deciding whether $[p[X_B]] = [q[X_B]]$ for $p[X_B]$ and $q[X_B]$ in $\mathbb{N}[X_B]$. In a nutshell, with each polynomial $p[X_B]$ in $\mathbb{N}[X_B]$, we first associate a special representative $p_e[X_B]$ in its equivalence class $[p[X_B]]$, called the *expanded version of* $p[X_B]$. Then, we show that $[p[X_B]] = [q[X_B]]$ if and only if $p_e[X_B]$ is "almost the same" as $q_e[X_B]$. We make this more precise later. The expanded versions are also shown to be useful for identifying representatives in the sum and product of equivalence classes.

6.2.1. The Expanded Version of a Polynomial. Intuitively, the expanded version of a polynomial $p[X_B]$ corresponds to an equivalent representation of $p[X_B]$ in terms of a set of basic, elementary polynomials in $\mathbb{N}[X_B]$. To define these basic elements, we first introduce some notation. We denote by $\mathcal{P}N$ the set of all pairs (P, N) where $P = \{\mu_1, \ldots, \mu_k\}$ and $N = \{\nu_1, \ldots, \nu_\ell\}$ are two multisets with elements from $mon(X)$. Furthermore, given $(P, N) \in \mathcal{P}N$ we denote by $\mathbf{b}_{P,N}$ the expression

$$b_{\mu_1} \cdot b_{\mu_2} \cdots b_{\mu_k} \cdot \bar{b}_{\nu_1} \cdot \bar{b}_{\nu_2} \cdots \bar{b}_{\nu_\ell}.$$

Example 6.2. Let $X = \{x_1, x_2\}$ and consider $P = \{x_1^2, x_2\}$ and $N = \{x_1, x_2, x_1x_2^3\}$. Then, we have that $\mathbf{b}_{P,N} = b_{x_1^2} b_{x_2} \bar{b}_{x_1} \bar{b}_{x_2} \bar{b}_{x_1x_2^3}$. Similarly, for $P' = \{x_1^2, x_2, x_2\}$ and $N' = \{x_1, x_1x_2^3\}$, the corresponding polynomial is given by $\mathbf{b}_{P',N'} = b_{x_1^2} b_{x_2}^2 \bar{b}_{x_1} \bar{b}_{x_1x_2^3}$. As another example, consider $P'' = \{x_1^2, x_2\}$ and $N'' = \{x_1, x_1x_2^3\}$. In this case, $\mathbf{b}_{P'',N''} = b_{x_1^2} b_{x_2} \bar{b}_{x_1} \bar{b}_{x_1x_2^3}$. Finally, for $P''' = \{x_1, x_1^2, x_2\}$ and $N''' = \{x_1x_2^3\}$ we have that $\mathbf{b}_{P''',N'''} = b_{x_1} b_{x_1^2} b_{x_2} \bar{b}_{x_1x_2^3}$.

When considering equivalence classes $[\mathbf{b}_{P,N}]$, we recall that "$\cdot$" is idempotent for variables in B_X and $\bar{B}_X$. In other words, $[\mathbf{b}_{P,N}] = [\mathbf{b}_{s(P),s(N)}]$ where $s(P)$ and $s(N)$ are the "set versions" of the multisets P and N, obtained by ignoring multiplicities of the monomials.

Example 6.3. Observe that $P' = \{x_1^2, x_2, x_2\}$ is the only proper multiset in the previous example. Clearly, $[\mathbf{b}_{P',N'}] = [b_{x_1^2} b_{x_2} \bar{b}_{x_1} \bar{b}_{x_1x_2^3}] = [\mathbf{b}_{s(P'),s(N')}]$.

Furthermore, since for any monomial μ, $[b_\mu \cdot \bar{b}_\mu] = [0]$, we may conclude that $[\mathbf{b}_{P,N}] = [0]$ whenever $P \cap N \neq \emptyset$.

Example 6.4. Since $P \cap N = \{x_2\}$ in Example 6.2, we have that $[\mathbf{b}_{P,N}] = [0]$.

This motivates the following definition.

Definition 6.5. Let S be a set of monomials. We say that $(P, N) \in \mathcal{P}N$ *represents an* S*-complete monomial* $\mathbf{b}_{P,N}$ if the following conditions are satisfied: (a) P and N are sets, (b) $S = P \cup N$, and (c) $P \cap N = \emptyset$.

Intuitively, when $(P, N) \in \mathcal{P}N$ represents an S-complete monomial, then every monomial $\mu \in S$ occurs once in $\mathbf{b}_{P,N}$ either as b_μ or $\bar{b}_\mu$.

Example 6.6. The pairs (P'', N'') and (P''', N''') in Example 6.2 represent the S-complete monomials $\mathbf{b}_{P'',N''} = b_{x_1^2} b_{x_2} \bar{b}_{x_1} \bar{b}_{x_1 x_2^3}$ and $\mathbf{b}_{P''',N'''} = b_{x_1} b_{x_1^2} b_{x_2} \bar{b}_{x_1 x_2^3}$, respectively, for $S = \{x_1, x_1^2, x_2, x_1 x_2^3\}$.

Finally, we denote by $\mathcal{C}(S)$ the set of pairs (P, N) that represent *non-zero* S-complete monomials (i.e., S-complete monomials for which $[\mathbf{b}_{P,N}] \neq [0]$). Note that it follows from rules (i) and (iv) that $(P, N) \in \mathcal{C}(S)$ if and only if (P, N) represents an S-complete monomial, and, moreover, for every $\mu \in P$ there does not exist a $\nu \in N$ such that $\mu = \nu \cdot \nu'$ for some monomial ν'.

Example 6.7. Continuing with the previous example, we have that $(P'', N'') \notin \mathcal{C}(S)$. Indeed, note that $x_1^2 \in P''$, $x_1 \in N''$ and $x_1^2 = x_1 \cdot x_1$. From this, it follows that $[b_{x_1^2} \bar{b}_{x_1}] = [0]$ and thus $[\mathbf{b}_{P'',N''}] = [0]$. By contrast, $(P''', N''') \in \mathcal{C}(S)$.

The polynomials $\mathbf{b}_{P,N}$ for $(P, N) \in \mathcal{C}(S)$ will play the role of the basic elements used to define the expanded versions of polynomials in $\mathbb{N}[X_B]$.

These elements exhibit the following useful properties, as shown in Lemma 6.8. The lemma is easily verified from the properties of the equivalence relation θ defined earlier, and its proof is deferred to the appendix.

LEMMA 6.8. *For any $(P, N) \in \mathcal{C}(S)$ and (P', N') whose monomials belong to S, we have that*

$$[\mathbf{b}_{P,N}] \cdot [\mathbf{b}_{P',N'}] = \begin{cases} [\mathbf{b}_{P,N}] & \text{if } s(P') \subseteq P \text{ and } s(N') \subseteq N \\ [0] & \text{otherwise.} \end{cases} \tag{1}$$

$$\left[\sum_{(P,N) \in \mathcal{C}(S)} \mathbf{b}_{P,N} \right] = [1], \tag{2}$$

where $s(P')$ and $s(N')$ denote the "set versions" of P' and N', respectively. In particular, if (P', N') also belongs to $\mathcal{C}(S)$, then $[\mathbf{b}_{P,N}] \cdot [\mathbf{b}_{P',N'}] = [\mathbf{b}_{P,N}]$ if $P = P'$ and $N = N'$; and $[\mathbf{b}_{P,N}] \cdot [\mathbf{b}_{P',N'}] = [0]$ otherwise.

Given a polynomial $p[X_B]$ in $\mathbb{N}[X_B]$, we next identify the set $S_{p[X_B]}$ of monomials that are relevant for obtaining the expanded version of $p[X_B]$. More precisely, the expanded version will be constructed in terms of $S_{p[X_B]}$-complete monomials.

Definition 6.9. Let $p[X_B]$ be a polynomial in $\mathbb{N}[X_B]$. If $p[X_B] = a$ for some constant $a \in \mathbb{N}$, then we define $S_{p[X_B]} := \{1\}$ where 1 is the constant monomial $x_1^0 \cdots x_n^0$. Otherwise, if $p[X_B]$ is not constant, then we define $S_{p[X_B]}$ as the set of all nonconstant monomials $\mu \in mon(X)$ that appear in $p[X_B]$, either as μ, $b_{\mu,}$ or $\bar{b}_\mu$. We will often abbreviate $S_{p[X_B]}$ and simply write S_p instead.

Example 6.10. For $p[X_B] = \bar{b}_{x_1 x_2} x_1 + b_{x_1} \bar{b}_{x_1} x_2^2 + 4 \bar{b}_{x_1} b_{x_2}^4 x_1 x_2 + \bar{b}_{x_1 x_2} x_2$, $S_p = \{x_1, x_2, x_2^2, x_1 x_2\}$, for $q[X_B] = x_2 + 3$, $S_q = \{x_2\}$, for $r[X_B] = x_2$, $S_r = \{x_2\}$ and for $t[X_B] = 5$, $S_t = \{1\}$.

We are now ready to define and construct the expanded version, denoted by $p_e[X_B]$, of a polynomial $p[X_B]$. We first illustrate this construction by means of an example. The general treatment is part of the proof of Proposition 6.14, where in addition the uniqueness of the resulting polynomial $p_e[X_B]$ will be established.

Example 6.11. Let $X = \{x_1, x_2\}$ and consider $p[X_B] = \bar{b}_{x_1x_2}x_1 + b_{x_1}\bar{b}_{x_1}x_2^2 + 4\bar{b}_{x_1}b_{x_2}^4x_1x_2 + \bar{b}_{x_1x_2}x_2$. We saw in the previous example that $S_p = \{x_1, x_2, x_2^2, x_1x_2\}$. We rewrite $p[X_B]$ as an equivalent polynomial in terms of S_p-complete polynomials. Observe that $\mathcal{C}(S_p)$ corresponds to the following $\mathbf{b}_{P,N}$'s:

$$b_{x_1}b_{x_2}b_{x_2^2}b_{x_1x_2},\ b_{x_1}b_{x_2}\bar{b}_{x_2^2}b_{x_1x_2},\ b_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2},\ b_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2},\ b_{x_1}\bar{b}_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2},$$
$$\bar{b}_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2},\ \bar{b}_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2},\ \bar{b}_{x_1}\bar{b}_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}.$$

Step 1. We first multiply $p[X_B]$ with $\sum_{(P,N)\in\mathcal{C}(S_p)}\mathbf{b}_{P,N}$. Observe that this results in an equivalent polynomial by Property (2) in Lemma 6.8. Using Property (1) in Lemma 6.8, we then eliminate all terms that are equivalent to 0 and further simplify each term such that every term consists of a basic element $\mathbf{b}_{P,N}$ and a polynomial in $\mathbb{N}[X]$. It is easily verified that the resulting equivalent polynomial is given by

$$\begin{aligned}p_1[X_B] = {} & b_{x_1}b_{x_2}b_{x_2^2}b_{x_1x_2}\cdot 0 + b_{x_1}b_{x_2}\bar{b}_{x_2^2}b_{x_1x_2}\cdot 0 + b_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2)\\ & + b_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2) + b_{x_1}\bar{b}_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2) + \bar{b}_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2+4x_1x_2)\\ & + \bar{b}_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2+4x_1x_2) + \bar{b}_{x_1}\bar{b}_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2),\end{aligned}$$

where we grouped terms according to their preceding basic element $\mathbf{b}_{P,N}$. We included the first two terms to illustrate that not all elements in $\mathcal{C}(S_p)$, when multiplied with $p[X_B]$, result in a non-zero term. For example, for the first term, we have $P = \{x_1, x_2, x_2^2, x_1x_2\}$ and $N = \emptyset$, whereas each of the terms in $p[X_B]$ contains an element of the form $\bar{b}_\mu$ for $\mu \in S_p$. Hence, by Property (1) in Lemma 6.8, $\mathbf{b}_{P,N}\cdot p[X_B]$ is equivalent to 0 for $P = \{x_1, x_2, x_2^2, x_1x_2\}$ and $N = \emptyset$. Similarly for the second zero term in $p_1[X_B]$.

Step 2. Although we have already eliminated some zero terms in $p_1[X_B]$ due to Property (1) in Lemma 6.8, we can further simplify $p_1[X_B]$ by removing from each term those monomials μ that appear together with $\bar{b}_\mu$. Note that, by construction, each monomial μ either appears together with b_μ or $\bar{b}_\mu$. The resulting equivalent polynomial is the expanded version $p_e[X_B]$ of $p[X_B]$ and is given by

$$\begin{aligned}p_e[X_B] = {} & b_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2) + b_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2) + b_{x_1}\bar{b}_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}x_1\\ & + \bar{b}_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2}x_2 + \bar{b}_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}x_2 + \bar{b}_{x_1}\bar{b}_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}\cdot 0,\end{aligned}$$

where we included the last zero term to illustrate that $\bar{b}_{x_1}\bar{b}_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2)$ is eliminated from $p_1[X_B]$. Also note that some terms have been simplified. For example, $\bar{b}_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}(x_1+x_2+4x_1x_2)$ in $p_1[X_B]$ has become $\bar{b}_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}x_2$. The set of remaining S_p-complete monomials $\mathbf{b}_{P,N}$ that occur in $p_e[X_B]$ together with a non-zero polynomial in $\mathbb{N}[X]$ is referred to as the S_p-support of $p[X_B]$.

The support of a polynomial in terms of S-complete monomials, for arbitrary sets S of monomials, is formally defined as follows.

Definition 6.12. Let $p[X_B]$ be a polynomial in $\mathbb{N}[X_B]$ and let S be a subset of $mon(X)$. The *S-support* of $p[X_B]$ is the set

$$S\text{-supp}(p[X_B]) := \{(P,N) \in \mathcal{C}(S) \mid [\mathbf{b}_{P,N}\cdot p[X_B]] \neq [0]\}.$$

Observe that two equivalent polynomials $p[X_B]$ and $q[X_B]$ have the same S-support. Indeed, $[p[X_B]] = [q[X_B]]$ implies that $[\mathbf{b}_{P,N}\cdot p[X_B]] = [\mathbf{b}_{P,N}\cdot q[X_B]]$ for any $(P,N) \in \mathcal{C}(S)$. Hence, $[\mathbf{b}_{P,N}\cdot p[X_B]] = [0]$ if and only if $[\mathbf{b}_{P,N}\cdot q[X_B]] = [0]$.

Example 6.13. Continuing with the previous example, the S_p-support of $p[X_B]$ is thus given by $\{(P_1, N_1), (P_2, N_2), (P_3, N_3), (P_4, N_4), (P_5, N_5)\}$ with $P_1 = \{x_1, x_2, x_2^2\}$, $N_1 = \{x_1x_2\}$, $P_2 = \{x_1, x_2\}$, $N_2 = \{x_2^2, x_1x_2\}$, $P_3 = \{x_1\}$, $N_3 = \{x_2, x_2^2, x_1x_2\}$, $P_4 = \{x_2, x_2^2\}$, $N_4 = \{x_1, x_1x_2\}$, $P_5 = \{x_2\}$, and $N_5 = \{x_1, x_2^2, x_1x_2\}$. Consider $p_e[X_B]$ in the previous example and denote the polynomials in $\mathbb{N}[X]$ accompanying the $\mathbf{b}_{P,N}$'s by $r_{P_1,N_1}[X] = r_{P_2,N_2}[X] = x_1 + x_2$, $r_{P_3,N_3}[X] = x_1$, and $r_{P_4,N_4}[X] = r_{P_5,N_5}[X] = x_2$, respectively. Then, $p_e[X_B]$ is equal to

$$p_e[X_B] = \sum_{(P,N)\in S_p\text{-supp}(p[X_B])} \mathbf{b}_{P,N} \cdot r_{P,N}[X],$$

where each $r_{P,N}[X] \in \mathbb{N}[X]$ is non-zero and such that none of its monomials appears in N.

The intuition and examples just described lead to the following proposition whose statement contains the formal definition of expanded version.

PROPOSITION 6.14. *Let $p[X_B]$ be a polynomial in $\mathbb{N}[X_B]$. Then, $[p[X_B]] = [p_e[X_B]]$, where*

$$p_e[X_B] = \sum_{(P,N)\in S_p\text{-}supp(p[X_B])} \mathbf{b}_{P,N} \cdot r_{P,N}[X], \qquad (\ddagger)$$

for some non-zero $r_{P,N}[X] \in \mathbb{N}[X]$ none of whose monomials appears in N. Furthermore, $p_e[X_B]$ is uniquely determined by $p[X_B]$.

Definition 6.15. The *expanded version* of $p[X_B]$ is the polynomial $p_e[X_B]$ given in (‡) in the statement of Proposition 6.14.

Observe that the expanded version can be of exponential size in the number of variables in S_p. The proof of Proposition 6.14 closely follows the construction given in Example 6.11 and is deferred to the appendix.

6.2.2. Testing Equivalence of Polynomials. The expanded versions are particularly useful when testing for equivalence. One may think that $[p[X_B]] = [q[X_B]]$ if and only if they have identical expanded versions $p_e[X_B] = q_e[X_B]$. This is not the case, however, simply because S_p can be different from S_q, and therefore p_e and q_e are expressed in terms of different complete monomials: S_p-complete monomials for p_e and S_q-complete monomials for q_e. Nevertheless, one can test equivalence of two polynomials by using expanded versions built from $(S_p \cup S_q)$-complete monomials. Indeed, in both of these expansions, the $(S_p \cup S_q)$-complete monomials must carry identical polynomials in $\mathbb{N}[X]$. The following proposition provides a necessary and sufficient condition for testing the equivalence of two polynomials in $\mathbb{N}[X_B]$.

PROPOSITION 6.16. *Let $p[X_B]$ and $q[X_B]$ be two polynomials in $\mathbb{N}[X_B]$ and let $p_e[X_B] = \sum_{(P,N)\in S_p\text{-supp}(p[X_B])} \mathbf{b}_{P,N} \cdot r_{P,N}[X]$ and $q_e[X_B] = \sum_{(P,N)\in S_q\text{-supp}(q[X_B])} \mathbf{b}_{P,N} \cdot s_{P,N}[X]$ be the expanded versions of $p[X_B]$ and $q[X_B]$, respectively. Then, $[p[X_B]] = [q[X_B]]$ if and only if for every (P, N) in $(S_p \cup S_q)$-supp$(p[X_B])$, we have that $r_{P|_{S_p},N|_{S_p}}[X] = s_{P|_{S_q},N|_{S_q}}[X]$, where $P|_{S_p}$, $N|_{S_p}$, $P|_{S_q}$ and $N|_{S_q}$ denote the restriction of P and N to S_p and S_q, respectively.*

We defer the proof of the proposition to the appendix. The use of the proposition is illustrated in the following example.

Example 6.17. Let $X = \{x_1, x_2\}$ and consider the polynomial $p[X_B]$ from the previous example, and the polynomial $q[X_B] = \bar{b}_{x_1x_2}(x_1 + x_2)$. Note that $S_q = \{x_1, x_2, x_1x_2\}$. It is readily verified that $q_e[X_B] = b_{x_1}b_{x_2}\bar{b}_{x_1x_2}(x_1 + x_2) + b_{x_1}\bar{b}_{x_2}\bar{b}_{x_1x_2}x_1 + \bar{b}_{x_1}b_{x_2}\bar{b}_{x_1x_2}x_2$ and

that the S_q-support of $q[X_B]$ is given by $\{(P'_1, N'_1), (P'_2, N'_2), (P'_3, N'_3)\}$ with $P'_1 = \{x_1, x_2\}$, $N'_1 = \{x_1x_2\}$, $P'_2 = \{x_1\}$, $N'_2 = \{x_2, x_1x_2\}$, $P'_3 = \{x_2\}$ and $N'_3 = \{x_1, x_1x_2\}$. Let $s_{P'_1,N'_1}[X] = x_1+x_2$, $s_{P'_2,N'_2}[X] = x_1$ and $s_{P'_3,N'_3}[X] = x_2$. Then, $q_e[X_B] = \mathbf{b}_{P'_1,N'_1}s_{P'_1,N'_1}[X]+\mathbf{b}_{P'_2,N'_2}s_{P'_2,N'_2}[X]+\mathbf{b}_{P'_3,N'_3}s_{P'_3,N'_3}[X]$. To check whether $p[X_B]$ and $q[X_B]$ are equivalent, we verify the condition stated in the previous proposition. Note that $S_p \cup S_q = S_p$ in this example. Hence, we have to further expand the S_q-complete monomials to S_p-complete monomials. It is easily verified that $q_e[X_B]$ is equivalent to

$$b_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2}s_{P'_1,N'_1}[X] + b_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}s_{P'_1,N'_1}[X] + b_{x_1}\bar{b}_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}s_{P'_2,N'_2}[X] + \bar{b}_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2}s_{P'_3,N'_3}[X] + \bar{b}_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}s_{P'_3,N'_3}[X].$$

Recall that $p_e[X_B]$ is equal to

$$b_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2}(x_1 + x_2) + b_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}(x_1 + x_2) + b_{x_1}\bar{b}_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}x_1 + \bar{b}_{x_1}b_{x_2}b_{x_2^2}\bar{b}_{x_1x_2}x_2 + \bar{b}_{x_1}b_{x_2}\bar{b}_{x_2^2}\bar{b}_{x_1x_2}x_2.$$

We have thus obtained precisely the same polynomials, which are clearly equivalent. For completeness, we formally verify the conditions stated in Proposition 6.16. Recall the formulation $p_e[X_B] = \sum_{(P,N)\in S_p\text{-supp}(p[X_B])} \mathbf{b}_{P,N} \cdot r_{P,N}[X]$ of $p_e[X_B]$ from Example 6.13. Observe that $P_1|_{S_q} = P_2|_{S_q} = P'_1$, $N_1|_{S_q} = N_2|_{S_q} = N'_1$, $P_3|_{S_q} = P'_2$, $N_3|_{S_q} = N'_2$, $P_4|_{S_q} = P_5|_{S_q} = P_3,'$ and $N_4|_{S_q} = N_5|_{S_q} = N'_3$. Then, $p[X_B]$ and $q[X_B]$ are indeed equivalent since $r_{P_1,N_1}[X] = r_{P_2,N_2} = s_{P'_1,N'_1}[X] = x_1 + x_2$, $r_{P_3,N_3}[X] = s_{P'_2,N'_2}[X] = x_1$, and $r_{P_4,N_4}[X] = r_{P_5,N_5}[X] = s_{P'_3,N'_3}[X] = x_2$.

As another example, consider $p[X_B] = \bar{b}_{x_1x_2}x_1$ and $q[X_B] = x_2$. We have that $S_p = \{x_1, x_1x_2\}$ and $S_q = \{x_1, x_2\}$. Furthermore, $p_e[X_B] = b_{x_1}\bar{b}_{x_1x_2}x_1$ and $q_e[X_B] = b_{x_1}b_{x_2}x_2 + \bar{b}_{x_1}b_{x_2}x_2$. Let $P_1 = \{x_1\}$, $N_1 = \{x_1x_2\}$, $P'_1 = \{x_1, x_2\}$, $N'_1 = \emptyset$, $P'_2 = \{x_2\}$, $N'_2 = \{x_1\}$, $r_{P_1,N_1}[X] = x_1$, $s_{P'_1,N'_1}[X] = s_{P'_2,N'_2}[X] = x_2$. Let $S = S_p \cup S_1 = \{x_1, x_2, x_1x_2\}$. We have that $p_e[X_B]$ is equivalent to

$$b_{x_1}b_{x_2}\bar{b}_{x_1x_2}r_{P_1,N_1}[X] + b_{x_1}\bar{b}_{x_2}\bar{b}_{x_1x_2}r_{P_1,N_1}[X],$$

and $q_e[X_B]$ is equivalent to

$$b_{x_1}b_{x_2}b_{x_1x_2}s_{P'_1,N'_1}[X] + b_{x_1}b_{x_2}\bar{b}_{x_1x_2}s_{P'_1,N'_1}[X] + \bar{b}_{x_1}b_{x_2}\bar{b}_{x_1x_2}s_{P'_2,N'_2}[X].$$

Consider $P = \{x_1, x_2, x_1x_2\}$ and $N = \emptyset$. Then, $P|_{S_p} = \{x_1, x_1x_2\}$, $N|_{S_p} = \emptyset$, $P|_{S_q} = \{x_1, x_2\} = P'_1$, and $N|_{S_q} = \emptyset = N'_1$. Since $p_e[X_B]$ does not have a term corresponding to $(\{x_1, x_1x_2\}, \emptyset)$ (in other words, $r_{(\{x_1,x_1x_2\},\emptyset)} = 0$), whereas $q_e[X_B]$ has a term, $r_{P'_1,N'_1}[X] = x_2$, corresponding to $(\{x_1, x_2\}, \emptyset)$, we may conclude that $p[X_B]$ and $q[X_B]$ are not equivalent.

One may wonder what the complexity is of deciding whether two polynomials are equivalent. The following proposition indicates that it is intractable, in general. The proof of the proposition is deferred to the appendix.

PROPOSITION 6.18. *Given two polynomials $p[X_B]$ and $q[X_B]$ in $\mathbb{N}[X_B]$, deciding whether $[p[X_B]] = [q[X_B]]$ holds is co*NP*-complete.*

6.2.3. Computing with Boolynomials. We conclude this section on boolynomials by showing that the expanded versions can also be used to compute representatives of sums and products of equivalence classes. The proof is omitted because it consists of a simple verification in which the expanded version of the sum and product of expanded versions are analyzed. Intuitively, when summing two boolynomials $p[X_B]$ and $q[X_B]$, one can simply sum their expanded versions, provided that one considers S-complete monomials where $S = S_p \cup S_q$. Moreover, the S-complete monomials of interest belong to the

union of the S-supports of $p[X_B]$ and $q[X_B]$. Similarly, multiplication can be expressed in terms of the product of the expanded version and the relevant S-monomials now belong to the *intersection* of the S-supports of p and q.

LEMMA 6.19. *Let $p[X_B]$ and $q[X_B]$ in $\mathbb{N}[X_B]$ and let $p_e[X_B]$ and $q_e[X_B]$ be their expanded versions. Let $S = S_p \cup S_q$. The expanded version of $p[X_B] + q[X_B]$ is equal to*

$$\sum_{(P,N)\in S\text{-supp}(p[X_B])\cup S\text{-supp}(q[X_B])} \mathbf{b}_{P,N} \cdot \left(r_{P|_{S_p},N|_{S_p}}[X] + s_{P|_{S_q},N|_{S_q}}[X]\right),$$

and the expanded version of $p[X_B] \cdot q[X_B]$ can be computed from

$$\sum_{(P,N)\in S\text{-supp}(p[X_B])\cap S\text{-supp}(q[X_B])} \mathbf{b}_{P,N} \cdot \left(r_{P|_{S_p},N|_{S_p}}[X] \cdot s_{P|_{S_q},N|_{S_q}}[X]\right),$$

by (i) eliminating all monomials μ in $r_{P|_{S_p},N|_{S_p}}[X] \cdot s_{P|_{S_q},N|_{S_q}}[X]$ that occur in N, and (ii) removing all $(P, N) \in S\text{-supp}(p[X_B]) \cap S\text{-supp}(q[X_B])$ for which the previous step completely eliminated $r_{P|_{S_p},N|_{S_p}}[X] \cdot s_{P|_{S_q},N|_{S_q}}[X]$.

We remark that it is not true in general that $S\text{-supp}(p[X_B] \cdot q[X_B]) = S\text{-supp}(p[X_B]) \cap S\text{-supp}(q[X_B])$, as is illustrated by the following example.

Example 6.20. Let $X = \{x_1, x_2\}$ and consider $p[X_B] = \bar{b}_{x_1x_2}x_1$ and $q[X_B] = x_2$ from the previous example. We established there that for $S = \{x_1, x_2, x_1x_2\}$, $S\text{-supp}(p[X_B])$ corresponds to $\{b_{x_1}b_{x_2}\bar{b}_{x_1x_2}, b_{x_1}\bar{b}_{x_2}\bar{b}_{x_1x_2}\}$ and $S\text{-supp}(q[X_B])$ corresponds to $\{b_{x_1}b_{x_2}b_{x_1x_2}, b_{x_1}b_{x_2}\bar{b}_{x_1x_2}, \bar{b}_{x_1}b_{x_2}\bar{b}_{x_1x_2}\}$. From the previous lemma, we may thus conclude that $b_{x_1}b_{x_2}b_{x_1x_2}x_2 + b_{x_1}b_{x_2}\bar{b}_{x_1x_2}(x_1 + x_2) + b_{x_1}\bar{b}_{x_2}\bar{b}_{x_1x_2}x_1 + \bar{b}_{x_1}b_{x_2}\bar{b}_{x_1x_2}x_2$ is the expanded version of $p[X_B] + q[X_B]$ and $S\text{-supp}(p[X_B] + q[X_B]) = S\text{-supp}(p[X_B]) \cup S\text{-supp}(q[X_B])$. Consider $S\text{-supp}(p[X_B]) \cap S\text{-supp}(q[X_B]) = \{b_{x_1}b_{x_2}\bar{b}_{x_1x_2}\}$. From the previous lemma, it follows that the expanded version of $p[X_B] \cdot q[X_B]$ can be computed by considering (i) $b_{x_1}b_{x_2}\bar{b}_{x_1x_2}x_1x_2$ and (ii) reducing this expression by eliminating all monomials μ that also appear as $\bar{b}_\mu$. In this case, $b_{x_1}b_{x_2}\bar{b}_{x_1x_2}x_1x_2$ is reduced to 0. Note that $S\text{-supp}(p[X_B] \cdot q[X_B])$ is empty, different from the intersection of the S-supports of $p[X_B]$ and $q[X_B]$.

6.3. Boolean Algebra

We next identify a boolean algebra that can be embedded in the semiring of boolynomials $\mathbb{N}_\theta[X_B]$. Intuitively, the elements in the boolean algebra represent the support of boolynomials (cf. Definition 6.12). Let $T \subseteq \mathcal{C}(S)$ for some $S \subset \mathit{mon}(X)$. That is, T represents a set of non-zero S-complete monomials. We call S the *carrier set* of T. Note that, given such a T, we can derive its carrier set S by simply collecting all monomials that appear in the elements $(P, N) \in T$. With each such T we associate the polynomial

$$p_T = \sum_{(P,N)\in T} \mathbf{b}_{P,N},$$

and define

$$\mathbb{B}_X := \{[p_T] \mid T \subseteq \mathcal{C}(S), S \subset \mathit{mon}(X)\} \subset \mathbb{N}_\theta[X_B].$$

We next define disjunction, conjunction, and complementation for the elements in $\mathbb{B}_X$. Let $[p_T]$ and $[p_{T'}]$ be elements in $\mathbb{B}_X$. Then,

$$[p_T] \vee_b [p_{T'}] := [p_T + p_{T'}] = [p_T] + [p_{T'}]$$
$$[p_T] \wedge_b [p_{T'}] := [p_T \cdot p_{T'}] = [p_T] \cdot [p_{T'}]$$
$$\overline{[p_T]}^b := [p_{\mathcal{C}(S)\setminus T}],$$

where S is the carrier set of T. In addition, we define $\bot_b$ and $\top_b$ as [0] and [1], respectively. It will sometimes be convenient to interpret $[0] = [\bar{b}_1]$ and $[1] = [b_1]$, which correspond to the constant polynomials $p_T = 0$ with $T = (P = \emptyset, N = \{1\})$ and $p_T = 1$ with $T = (P = \{1\}, N = \emptyset)$, respectively.

In the appendix, we verify that we indeed obtain a boolean algebra:

PROPOSITION 6.21. *The operations $\vee_b$, $\wedge_b$ and $^{-b}$ are well-defined and the structure $(\mathbb{B}_X, \vee_b, \wedge_b, ^{-b}, \bot_b, \top_b)$ is a boolean algebra.*

6.4. Universal Seba-Structure

With the semiring of boolynomials $(\mathbb{N}_\theta[X_B], +, \cdot, 0, 1)$ and boolean algebra $(\mathbb{B}_X, \vee_b, \wedge_b, ^{-b}, \bot_b, \top_b)$ at hand, we next link them together by means of mappings $\iota_b : \mathbb{B}_X \to \mathbb{N}_\theta[X_B]$ and $d_b : \mathbb{N}_\theta[X_B] \to \mathbb{B}_X$. For ι_b, we simply take the identity mapping. That is, for $[p_T] \in \mathbb{B}_X$ we define

$$\iota_b([p_T]) := [p_T].$$

The mapping d_b is defined as follows. Let $p[X_B]$ be a polynomial in $\mathbb{N}[X_B]$. We define

$$d_b([p[X_B]]) := \left[p_{S_p\text{-supp}(p[X_B])}\right],$$

where S_p is as defined in Definition 6.9. Note that $d_b([0]) = [p_{\{x_1\}\text{-supp}(0)}] = [p_\emptyset] = [0]$ and $d_b([n]) = [p_{\{x_1\}\text{-supp}(n)}] = [p_{\mathcal{C}(\{x_1\})}] = [1]$.

The mapping d_b is well-defined. Indeed, given $[p[X_B]] = [q[X_B]]$, we saw that $S\text{-supp}(p[X_B]) = S\text{-supp}(q[X_B])$ for any $S \subseteq mon(X)$. Moreover, it is easily verified that, for $S_p \subseteq S \subseteq S'$, $[p_{S\text{-supp}(p[X_B])}] = [p_{S'\text{-supp}(p[X_B])}]$. Hence, $d_b([p[X_B]])$ is equal to

$$\left[p_{S_p\text{-supp}(p[X_B])}\right] = \left[p_{(S_p \cup S_q)\text{-supp}(p[X_B])}\right] = \left[p_{(S_p \cup S_q)\text{-supp}(q[X_B])}\right] = \left[p_{S_q\text{-supp}(q[X_B])}\right],$$

which equals $d_b([q[X_B]])$.

We conclude this section by showing that $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$ is indeed a seba-structure and that it is universal.

THEOREM 6.22. *The structure $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$ is a seba-structure that is universal in the class of all seba-structures, relative to the generator set X.*

PROOF. We defer the proof that $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$ is a seba-structure to the appendix and focus here on the universality of $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$.

Let (K, B, d, ι) be a seba-structure consisting of a semiring $(K, \oplus_K, \otimes_K, 0_K, 1_K)$, boolean algebra $(B, \vee, \wedge, \bar{}, \bot, \top)$, and mappings $d{:}K \to B$ and $\iota{:}B \to K$. Let $X = \{x_1, \ldots, x_n\}$ be a set of variables and denote by $[X]$ the set of equivalence classes $[x_i]$, for $x_i \in X$. Let $\nu : [X] \to K$ be a valuation that assigns values in K to the equivalence classes of variables in X.

We establish the universality of $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$ in the standard way; that is, we assume that we have a seba-homomorphism (h, β) from $(\mathbb{N}_b[X_B], \mathbb{B}_X, d_b, \iota_b)$ to (K, B, d, ι) and then show that it is uniquely determined by the given mapping $\nu : [X] \to K$.

Let $p[X_B] \in \mathbb{N}[X_B]$. We may assume that $p[X_B] = \sum_{(P,N)\in T} \mathbf{b}_{P,N} \cdot p_{P,N}[X]$ for some $T \subseteq \mathcal{P}N$ and $p_{P,N}[X] = \sum_\mu a_{P,N,\mu} \cdot \mu$, where $\mu \in mon(X)$ and $a_{P,N,\mu} \in \mathbb{N}$. Since equivalence classes are compatible with addition and multiplication, we can write

$$[p[X_B]] = \sum_{(P,N)\in T} [\mathbf{b}_{P,N}] \cdot [p_{P,N}[X]].$$

Furthermore, $[\mathbf{b}_{P,N}] = \prod_{\mu\in P}[b_\mu]\cdot\prod_{\mu\in N}[\bar{b}_\mu]$ and $[p_{P,N}[X]] = \sum_\mu [a_{P,N,\mu}]\cdot[\mu]$. Since h is assumed to be a semiring-homomorphism, we must have that

$$h([p[X_B]]) = \sum_{(P,N)\in T}\prod_{\mu\in P} h([b_\mu])\cdot\prod_{\mu\in N} h([\bar{b}_\mu])\cdot\left(\sum_\mu h([a_{P,N,\mu}])\cdot h([\mu])\right).$$

Let us consider first $h([a_{P,N,\mu}])$. Denote by $([\mathbb{N}], +, \cdot, [0], [1])$ the semiring obtained as the quotient of $(\mathbb{N}, +, \cdot, 0, 1)$ over the congruence relation θ. It is readily verified that if h, when restricted to elements in $[\mathbb{N}]$, is a semiring homomorphism from $[\mathbb{N}]$ to K, then h must coincide with $\kappa : [\mathbb{N}] \to K$, the canonical semiring morphism defined as $\kappa([0]) = 0_K$, $\kappa([1]) = 1_K$ and for $m > 1$, $\kappa([m]) = \underbrace{1_K \oplus \cdots \oplus 1_K}_{m \text{ times}}$.

Consider next the expression $h([\mu])$ for $\mu = x_1^{i_1}\cdot x_2^{i_2}\cdots x_n^{i_n}$ with $i_j \in \mathbb{N}$ for $j \in [1, n]$. The requirement that h is a semiring homomorphism and must coincide with $\nu : [X] \to K$ implies that

$$\begin{aligned} h([\mu]) &= h\left(\left[x_1^{i_1}\right]\right)\otimes\cdots\otimes h\left(\left[x_n^{i_n}\right]\right)\\ &= h([x_1]^{i_1})\otimes\cdots\otimes h([x_n]^{i_n})\\ &= (h([x_1]))^{i_1}\otimes\cdots\otimes(h([x_n]))^{i_n}\\ &= (\nu([x_1]))^{i_1}\otimes\cdots\otimes(\nu([x_n]))^{i_n}. \end{aligned}$$

Hence, $h([\mu])$ is fully determined by $\nu : [X] \to K$.

It remains to identify $h([b_\mu])$ and $h([\bar{b}_\mu])$. Observe that $[b_\mu] = \imath_b([b_\mu]) = \imath_b(d_b([\mu]))$ and $[\bar{b}_\mu] = \imath_b([\bar{b}_\mu]) = \imath_b(\overline{d_b([\mu])}^b)$. Since (h, β) must be a seba-homomorphism, we have that $h([b_\mu]) = h(\imath_b(d_b([\mu]))) = \imath(\beta(d_b([\mu]))) = \imath(d(h([\mu])))$ which, as just shown, is fully determined by $\nu : [X] \to K$. Similarly, $h([\bar{b}_\mu]) = h(\imath_b(\overline{d_b([\mu])}^b)) = \imath(\beta(\overline{d_b([\mu])}^b)) = \imath(\overline{\beta(d_b([\mu]))}^b) = \imath(\overline{d(h([\mu]))})$, which is again fully determined by $\nu : [X] \to K$. Here, we use that β is compatible with complementation.

In other words, $h([p[X_B]])$ is uniquely determined by ν and the requirement that (h, β) is a seba-homomorphism. Indeed, we have shown that $h([p[X_B]])$ is given by

$$\bigoplus_{(P,N)\in T}\left(\bigotimes_{\mu\in P}\imath(d(\nu([\mu])))\otimes\bigotimes_{\mu\in N}\imath(\overline{d(\nu([\mu]))})\right)\otimes\left(\bigoplus_\mu \kappa([a_{P,N,\mu}])\otimes\nu([\mu])\right),$$

where $\kappa : [\mathbb{N}] \to K$ is the homomorphism previously introduced and $\nu([\mu]) = (\nu([x_1]))^{i_1}\otimes\cdots\otimes(\nu([x_n]))^{i_n}$ for $\mu = x_1^{i_1}\cdot x_2^{i_2}\cdots x_n^{i_n}$ with $i_j \in \mathbb{N}$ for $j \in [1, n]$.

Observe that we did not yet define $\beta([p_T])$. A similar reasoning as for h shows that, for an element $[p_T] = [\sum_{(P,N)\in T}\mathbf{b}_{P,N}]$,

$$\beta([p_T]) = \bigvee_{(P,N)\in T}\bigwedge_{\mu\in P} d(h([\mu]))\wedge\bigwedge_{\mu\in N}\overline{d(h([\mu]))}.$$

To conclude the proof, we show in the appendix that (h, β) is well-defined (i.e., it does not depend on the chosen representative of the equivalence classes). Furthermore, it is easy to verify that (h, β) is a seba-homomorphism. Indeed, this can be verified by showing that h is a semiring homomorphism, that β commutes with complementation, and, in addition, $\beta(d_b(k)) = d(h(k))$ and $h(\imath_b(b)) = \imath(\beta(b))$ holds. We verify these latter two commutation conditions in the appendix.

6.5. Universal Spm-Semiring

Having a universal seba-structure at our disposal, we next describe the corresponding universal spm-semiring. The following lemma relates seba-homomorphisms and spm-semiring homomorphisms.

Let $(K, \oplus_K, \otimes_K, \ominus_K, 0_K, 1_K)$ and $(L, \oplus_L, \otimes_L, \ominus_L, 0_L, 1_L)$ be two spm-semirings. A *homomorphism* between these structures is a mapping $h : K \to L$ such that $h(0_K) = 0_L$ and $h(1_K) = 1_L$, $h(x \oplus_K y) = h(x) \oplus_L h(y)$, $h(x \otimes_K y) = h(x) \otimes_L h(y)$, and $h(x \ominus_K y) = h(x) \ominus_L h(y)$.

LEMMA 6.23. *A seba-homomorphism between two seba-structures induces a homomorphism between their derived spm-semirings.*

PROOF. Let (h, β) be a seba-homomorphism from (K_1, B_1, d_1, ι_1) to (K_2, B_2, d_2, ι_2). We extend K_1 and K_2 with the derived minus operators $k \ominus_1 \ell = k \otimes_1 \iota_1(\overline{d_1(\ell)}^1)$ and $k' \ominus_2 \ell' = k' \otimes_2 \iota_2(\overline{d_2(\ell')}^2)$, for $k, \ell \in K_1$ and $k', \ell' \in K_2$. We show that $h(k \ominus_1 \ell) = h(k) \ominus_2 h(\ell)$. Observe that $h(k \ominus_1 \ell) = h(k \otimes_1 \iota_1(\overline{d_1(\ell)}^1)) = h(k_1) \otimes_2 h(\iota_1(\overline{d_1(\ell)}^1))$. Furthermore, $h(\iota_1(\overline{d_1(\ell)}^1)) = \iota_2(\beta(\overline{d_1(\ell)}^1)) = \iota_2(\overline{\beta(d_1(\ell))}^2) = \iota_2(\overline{d_2(h(\ell))}^2)$. It follows that $h(k \ominus_1 \ell) = h(k \otimes_1 \iota_1(\overline{d_1(\ell)}^1)) = h(k) \otimes_2 \iota_2(\overline{d_2(h(\ell))}^2) = h(k) \ominus_2 h(\ell)$.

We are now finally ready to define the universal spm-semiring. An spm-semiring $(K, \oplus_K, \otimes_K, \ominus_K, 0_K, 1_K)$ is *universal* in the class of all spm-semirings, relative to a set of generators $X = \{x_1, \ldots, x_n\}$, if for any spm-semiring $(L, \oplus_L, \otimes_L, \ominus_L, 0_L, 1_L)$ and any valuation $\nu : X \to L$ that assigns values from L to variables in X, we can *uniquely* extend ν to an spm-semiring homomorphism $h : K \to L$ such that h coincides with ν on X.

We next introduce the spm-semiring derived from the universal seba-structure $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$. □

Definition 6.24. We define the spm-semiring $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ as the semiring of boolynomials $(\mathbb{N}_\theta[X_B], +, \cdot, 0, 1)$ equipped with

$$[p[X_B]] - [q[X_B]] := [p[X_B]] \cdot [p_{\mathcal{C}(S_q) \setminus S_q}],$$

where S_q is the support of $q[X_B]$.

PROPOSITION 6.25. *The spm-semiring* $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ *is universal in the class of spm-semirings.*

PROOF. Consider an spm-semiring $(K, \oplus, \otimes, \ominus, 0, 1)$ and valuation $\nu : [X] \to K$. We define the spm-semiring homomorphism h_s from $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ to K in terms of the unique seba-homomorphism (h, β) from the seba-structure $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$ to the seba-structure (K, B, d, ι) from which $(K, \oplus, \otimes, \ominus, 0, 1)$ is derived as constructed in the proof of Theorem 5.3. In particular, we define $h_s := h$. Lemma 6.23 implies that h_s is indeed an spm-homomorphism. Furthermore, Theorem 6.22 tells us that h_s is fully determined by the valuation ν. It remains to show that the definition of h_s is independent of the choice of seba-structure from which $(K, \oplus, \otimes, \ominus, 0, 1)$ is derived. Indeed, let (K, B_1, d_1, ι_1) and (K, B_2, d_2, ι_2) be two seba-structures from which $(K, \oplus, \otimes, \ominus, 0, 1)$ is derived. That is, for any $k, \ell \in K$,

$$k \ominus \ell = k \otimes \iota_1(\overline{d_1(\ell)}^1) = k \otimes \iota_2(\overline{d_2(\ell)}^2).$$

In particular, for $k = 1$, this implies that $\iota_1(\overline{d_1(\ell)}^1) = \iota_2(\overline{d_2(\ell)}^2)$ for any $\ell \in K$. Let (h_1, β_1) and (h_2, β_2) be the two seba-homomorphisms from $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$ to (K, B_1, d_1, ι_1) and (K, B_2, d_2, ι_2), respectively, both of which extend the valuation $\nu : [X] \to K$ as given in the proof of Theorem 6.22.

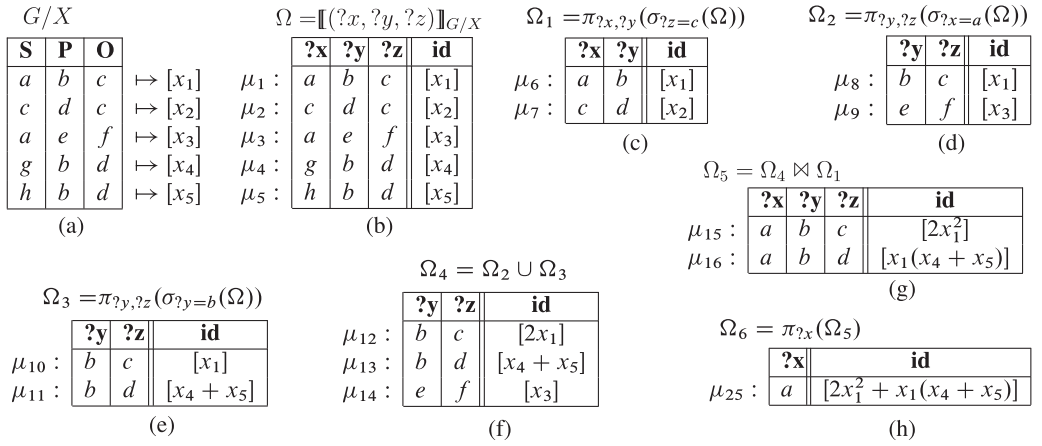

Fig. 3. Example of RDF graph and evaluation of positive SPARQL algebra operators with annotations in the universal spm-semiring.

Observe that the proof of Theorem 6.22 reveals that we only need to verify whether for each $\mu \in mon(X)$, $h_1([b_\mu]) = h_2([b_\mu])$ and $h_1([\bar{b}_\mu]) = h_2([\bar{b}_\mu])$. To see this, observe that $h_1([\bar{b}_\mu]) = h_1(\iota_b(\overline{d_b([\mu])}^b)) = \iota_1(\beta(\overline{d_b([\mu])}^b)) = \iota_1(\overline{\beta(d_b([\mu]))}^1) = \iota_1(\overline{d_1(h_1([\mu]))}^1) = \iota_2(\overline{d_2(h_1([\mu]))}^2)$ and that $h_1([\mu]) = h_2([\mu])$ because these are fully determined by ν. Hence, $\iota_2(\overline{d_2(h_1([\mu]))}^2) = \iota_2(\overline{d_2(h_2([\mu]))}^2)$ and thus $h_1([\bar{b}_\mu]) = h_2([\bar{b}_\mu])$. A similar argument, using that $[b_\mu] = \iota_b(\overline{\overline{d_b([\mu])}^b})$, shows that $h_1([b_\mu]) = h_2([b_\mu])$.

Hence, h_s is a well-defined spm-homomorphism that is fully determined by the valuation $\nu : [X] \to K$. Hence, $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ is indeed a universal spm-semiring. □

7. PROVENANCE AND FACTORIZATION PROPERTY

We next show how the universal spm-semiring $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ can be used to model the *how-provenance* of SPARQL query results. Similar to the relational case [Green et al. 2007], we start by introducing the abstractly tagged version of an RDF graph.

Definition 7.1. Let G be an RDF graph and let $[X] = \{[x_1], \ldots, [x_n]\}$ be a set of equivalence classes, one for each triple in G. The *abstractly tagged* version of G, denoted by G/X, is the $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotated RDF graph in which each triple (s, p, o) in G is annotated with its corresponding equivalence class in $[X]$.

Given G/X, we now illustrate how $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotated mapping sets corresponding to the results of positive SPARQL algebra expressions can be computed using the propagation rules given in Section 3.

Example 7.2. Figure 3(b) shows the $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotated mapping set $\Omega = \llbracket(?x, ?y, ?z)\rrbracket_{G/X}$ for the abstractly tagged version of the RDF graph given in Figure 3(a). Here, $X = \{x_1, \ldots, x_5\}$, and each triple is annotated with a unique equivalence class $[x_i]$. Recall the positive SPARQL algebra expressions described in Example 2.1 and shown in Figure 1. Figures 3(c)–(g) depict the corresponding $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotated mapping sets Ω_i, for $i \in [1, 5]$, respectively. In addition, Figure 3(h) shows the $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotated mapping set Ω_6 corresponding to $\pi_{?x}(\Omega_5)$. The annotations in these mapping sets are computed as defined in Section 3, in which the

$\Omega_4 \setminus \Omega_1$

	?y	?z	id
μ_{17} :	b	c	$[2x_1] - [x_1]$
μ_{18} :	b	d	$[x_4 + x_5] - [x_1]$
μ_{19} :	e	f	$[x_3]$

(a)

$(\Omega_4 \setminus \Omega_1) \setminus \Omega_6$

	?y	?z	id
μ_{26} :	b	c	$([2x_1] - [x_1]) - [2x_1^2 + x_1(x_4 + x_5)]$
μ_{27} :	b	d	$([x_4 + x_5] - [x_1]) - [2x_1^2 + x_1(x_4 + x_5)]$
μ_{28} :	e	f	$[x_3] - [2x_1^2 + x_1(x_4 + x_5)]$

(b)

$\Omega_4 \rtimes \Omega_1$

	?x	?y	?z	id
μ_{20} :	a	b	c	$[2x_1^2]$
μ_{21} :	a	b	d	$[x_1(x_4 + x_5)]$
μ_{22} :	–	b	c	$[2x_1] - [x_1]$
μ_{23} :	–	b	d	$[x_4 + x_5] - [x_1]$
μ_{24} :	–	e	f	$[x_3]$

(c)

$\Omega_4 \setminus (\Omega_1 \cup \Omega_6)$

	?y	?z	id
μ_{29} :	b	c	$[2x_1] - [x_1 + 2x_1^2 + x_1(x_4 + x_5)]$
μ_{30} :	b	d	$[x_4 + x_5] - [x_1 + 2x_1^2 + x_1(x_4 + x_5)]$
μ_{31} :	e	f	$[x_3] - [2x_1^2 + x_1(x_4 + x_5)]$

(d)

Fig. 4. Example of RDF graph and evaluation of non-monotone SPARQL algebra operators with annotations in the universal spm-semiring. Gray-shaded entries are not part of the support of the mapping sets because their boolynomials are equivalent to [0].

abstract operators $\oplus$ and $\otimes$ are replaced by the corresponding operators "+" and "$\cdot$" in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$, respectively. It is readily verified that when positive SPARQL queries are concerned, one can simply compute the provenance by means of equivalence classes $[p[X]]$ of standard polynomials $p[X]$ in $\mathbb{N}[X]$. Indeed, this follows from the properties of equivalence classes: For any pair of polynomials $p[X]$ and $q[X]$ in $\mathbb{N}[X]$, $[p[X] + q[X]] = [p[X]] + [q[X]]$ and $[p[X] \cdot q[X]] = [p[X]] \cdot [q[X]]$.

The previous example shows that when the positive algebra is concerned, the annotations in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ can be interpreted in precisely the same way as in the relational case: "+" corresponds to union and "$\cdot$" corresponds to join. For example, μ_{16} in Ω_5 is obtained by joining the triple with id $[x_1]$ with the union of the two triples with id's $[x_4]$ and $[x_5]$. We next consider the non-monotone operators $\setminus$ and $\rtimes$.

Example 7.3. Consider the SPARQL algebra expressions $\Omega_4 \setminus \Omega_1$, $(\Omega_4 \setminus \Omega_1) \setminus \Omega_6$, $\Omega_4 \rtimes \Omega_1$, and $\Omega_4 \setminus (\Omega_1 \cup \Omega_6)$. Figures 4(a), (b), (c), and (d) show the corresponding $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotated mapping sets, computed as defined in Section 3 in which the operator $\ominus$ is now replaced by the corresponding "–" operator in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$. Also in this case, the elements in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ represent the how-provenance of mappings. For example, $[x_4 + x_5] - [x_1]$ indicates that μ_{18} is obtained from the union of the triples with id's $[x_4]$ and $[x_5]$, from which the triple with id $[x_1]$ is "subtracted". In other words, the mappings μ_4 and μ_5 determine the provenance of μ_{18} as long as μ_1 is not present. The mapping μ_{18} is thus conditionally present in $\Omega_4 \setminus \Omega_1$. In some cases, however, we can eliminate certain mappings unconditionally. Let us consider the annotation $[2x_1] - [x_1]$ of the mapping μ_{17} in $\Omega_4 \setminus \Omega_1$. One could simply leave this annotation as it is, or one could benefit from the fact that $[2x_1] - [x_1]$ is an element in the spm-semiring $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$. Indeed, from id_{11}, we have that $2([x_1] - [x_1])$ represents the same element as $[2x_1] - [x_1]$, and id_9 implies that $2([x_1] - [x_1]) = 2[0] = [0]$. In other words, μ_{17} is not part of the support of $\Omega_4 \setminus \Omega_1$ and can thus be eliminated. Similarly, μ_{22} is not part of the support of $\Omega_4 \rtimes \Omega_1$. We showed in Proposition 4.4 that the identity $0 \ominus k = 0$ holds in spm-semirings K for any $k \in K$. Hence, the annotation $([2x_1] - [x_1]) - [2x_1^2 + x_1(x_4 + x_5)]$ of μ_{26} is equal to $[0]$ and hence, μ_{26} can be eliminated from $(\Omega_4 \setminus \Omega_1) \setminus \Omega_6$ as well. Finally, id_{10} implies that μ_{29} is equal

to 0. Indeed, $[2x_1] - [x_1 + 2x_1^2 + x_1(x_4 + x_5)]$ is equal to $([2x_1] - [x_1]) - [2x_1^2 + x_1(x_4 + x_5)]$, which we have just shown to be [0].

Identifying the support of $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotated mapping sets, as illustrated in the previous example, comes at a cost. Indeed, we recall from Proposition 6.18 that deciding equivalence in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ is coNP-complete. A similar argument as given in the proof of Proposition 6.18 shows that determining whether a mapping belongs to the support is NP-complete. Indeed, this requires checking whether a given element in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ is *not* equivalent to [0], which can be shown to be NP-complete. As a note aside, a similar situation occurs when negation is recorded in provenance models by means of propositional logic expressions [Geerts and Poggi 2010; Dividino et al. 2009]. Identifying the support in those settings also incurs the cost of testing the satisfiability of propositional logic expressions, which is known to be NP-complete [Garey and Johnson 1979].

In our more general setting, and as the previous example shows, one way of determining the support is by relying on the identities of spm-semirings (as shown in Figure 2). A more procedural way is by relying on the definition of "$-$" in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ as given in Definition 6.24 and verifying whether or not the expanded version in the equivalence classes is equal to 0 (cf. Proposition 6.16). We illustrate this with the following example.

Example 7.4. Consider again the annotation $[2x_1] - [x_1]$ of μ_{17} in $\Omega_4 \setminus \Omega_1$. By definition, this annotation is equivalent to $[2\bar{b}_{x_1}x_1]$. Since $[\bar{b}_\mu \mu] = [0]$, this implies that the expanded version of $[2\bar{b}_{x_1}x_1]$ is equal to [0]. As another example, consider the annotation $([x_4 + x_5] - [x_1]) - [2x_1^2 + x_1(x_4 + x_5)]$ of μ_{27}. Definition 6.24 tells us that this is equivalent to

$$([x_4 + x_5][\bar{b}_{x_1}])[p_{\mathcal{C}(S)\setminus T}],$$

where $S = \{x_1, x_1^2, x_4, x_5\}$ and T is the S-support of $2x_1^2 + x_1(x_4 + x_5)$. We previously illustrated that T is computed by first enumerating all S-complete monomials in $\mathcal{C}(S)$ and then verifying when their product with $2x_1^2 + x_1(x_4 + x_5)$ returns non-zero. In this example, $\mathcal{C}(S)$ consists of 12 monomials, which we next list together with their corresponding term in $2x_1^2 + x_1(x_4 + x_5)$:

$b_{x_1} b_{x_1^2} b_{x_4} b_{x_5}$	$2x_1^2 + x_1(x_4 + x_5)$	$b_{x_1} \bar{b}_{x_1^2} b_{x_4} \bar{b}_{x_5}$	$x_1 x_4$
$b_{x_1} \bar{b}_{x_1^2} b_{x_4} b_{x_5}$	$x_1(x_4 + x_5)$	$b_{x_1} b_{x_1^2} \bar{b}_{x_4} \bar{b}_{x_5}$	$2x_1^2$
$b_{x_1} b_{x_1^2} \bar{b}_{x_4} b_{x_5}$	$2x_1^2 + x_1 x_5$	$\bar{b}_{x_1} \bar{b}_{x_1^2} \bar{b}_{x_4} b_{x_5}$	0
$b_{x_1} b_{x_1^2} b_{x_4} \bar{b}_{x_5}$	$2x_1^2 + x_1 x_4$	$\bar{b}_{x_1} \bar{b}_{x_1^2} b_{x_4} \bar{b}_{x_5}$	0
$\bar{b}_{x_1} \bar{b}_{x_1^2} b_{x_4} b_{x_5}$	0	$b_{x_1} \bar{b}_{x_1^2} \bar{b}_{x_4} \bar{b}_{x_5}$	0
$b_{x_1} \bar{b}_{x_1^2} \bar{b}_{x_4} b_{x_5}$	$x_1 x_5$	$\bar{b}_{x_1} \bar{b}_{x_1^2} \bar{b}_{x_4} \bar{b}_{x_5}$	0

Hence, $p_{\mathcal{C}(S)\setminus T}$ corresponds to the boolynomial

$$[\bar{b}_{x_1} \bar{b}_{x_1^2} b_{x_4} b_{x_5} + \bar{b}_{x_1} \bar{b}_{x_1^2} \bar{b}_{x_4} b_{x_5} + \bar{b}_{x_1} \bar{b}_{x_1^2} b_{x_4} \bar{b}_{x_5} + b_{x_1} \bar{b}_{x_1^2} \bar{b}_{x_4} \bar{b}_{x_5} + \bar{b}_{x_1} \bar{b}_{x_1^2} \bar{b}_{x_4} \bar{b}_{x_5}].$$

When multiplied with $\bar{b}_{x_1}(x_4 + x_5)$ we obtain the non-zero expanded form of $([x_4 + x_5] - [x_1]) - [2x_1^2 + x_1(x_4 + x_5)]$:

$$[(\bar{b}_{x_1} \bar{b}_{x_1^2} b_{x_4} b_{x_5} + \bar{b}_{x_1} \bar{b}_{x_1^2} \bar{b}_{x_4} b_{x_5} + \bar{b}_{x_1} \bar{b}_{x_1^2} b_{x_4} \bar{b}_{x_5} + \bar{b}_{x_1} \bar{b}_{x_1^2} \bar{b}_{x_4} \bar{b}_{x_5}) \cdot (x_4 + x_5)].$$

From Proposition 6.16, we may thus conclude that μ_{27} carries an annotation different from [0] and thus belongs to the support of $(\Omega_4 \setminus \Omega_1) \setminus \Omega_6$.

We also remark that the use of an spm-semiring as annotation structure has the additional benefit that mappings generated by equivalent SPARQL algebra expressions carry the same element in the spm-semiring. Proposition 6.18 tells us that the process of identifying equivalent expressions is coNP-complete, however.

Example 7.5. We know from Section 4 that $(\Omega_4 \setminus \Omega_1) \setminus \Omega_6 \equiv \Omega_4 \setminus (\Omega_1 \cup \Omega_6)$ and that spm-semirings satisfy the corresponding identity id_{10}. Using this identity, we may conclude that annotations for mappings μ_{26} and μ_{29} are the same (we previously showed these to be [0]). Similarly for μ_{27} and μ_{30}, and μ_{28} and μ_{31}. We may again rely on either identities in spm-semirings or Proposition 6.16 to detect when two expressions are equivalent.

Despite the fact that deciding the support of $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotated mapping sets and deciding equivalence of $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotations is intractable, we emphasize that the incorporation of $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$-annotations during the SPARQL querying process has *poly-size overhead* compared to the unannotated setting, a desirable property of annotation models [Amsterdamer et al. 2011b]. This may seem counterintuitive since the expanded version of an expression in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ may result in an exponential blowup in the size of the provenance expressions. However, when provenance is concerned, there is no need to consider the expanded versions. Instead, one simply stays within the realm of the spm-semiring $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ by propagating the annotations along with the SPARQL operators as described in Section 3. It can easily be verified by induction on the structure of the SPARQL graph pattern P and by using the propagation rules from Section 3 that the representation size of $[\![P]\!]_{G_a}$, including annotations, is indeed polynomial in the number of triples in G_a and the maximal length ℓ of annotation in G_a. Of course, this only holds when the graph pattern P is assumed to be fixed and thus the size of P is constant—in other words, when *data complexity* is concerned [Vardi 1982]. Note that when *combined complexity* [Vardi 1982] is concerned, even deciding whether a mapping is in the result of a SPARQL query (in the standard unannotated setting) is already PSPACE-complete [Pérez et al. 2009], and the issue of having poly-size overhead is less of a concern.

If the expanded versions are not needed when provenance is concerned, one may wonder at this point why we do not define $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ simply as the spm-semiring obtained by (i) taking the symbolic expressions built up from variables in X, constants in $\mathbb{N}$, and operations $+$, $\cdot$, and $-$; and by (ii) considering equivalence classes determined by the spm-semiring identities as given in Figure 2. This is the standard construction for universal structures in equational varieties [Burris and Sankappanavar 1981]. Such an approach, however, does not provide a proper semantics of the equivalence classes. One would have to solely rely on the identities to determine equivalence of expressions or membership in the support. Furthermore, it is not clear at all how canonical representatives in the equivalence classes can be obtained. In our approach, the expanded versions of elements in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ are crucial to achieve all of this, whenever needed.

The intractability related to working with expanded versions does not mean that they are without merit in the context of provenance, in a similar way as propositional logic is used in many practical settings. We next illustrate this with two examples. First, we show how expanded versions can help to minimize provenance expressions in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$.

Example 7.6. Proposition 6.14 tells us that expanded versions provide a canonical representative of the equivalence classes in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$. One can often further simplify these expressions by applying rules similar to those used in boolean function

minimization. For instance, we saw in Example 7.4 that $([x_4 + x_5] - [x_1]) - [2x_1^2 + x_1(x_4 + x_5)]$ can be expressed equivalently as $[(\bar{b}_{x_1}\bar{b}_{x_1^2}b_{x_4}b_{x_5} + \bar{b}_{x_1}\bar{b}_{x_1^2}\bar{b}_{x_4}b_{x_5} + \bar{b}_{x_1}\bar{b}_{x_1^2}b_{x_4}\bar{b}_{x_5} + \bar{b}_{x_1}\bar{b}_{x_1^2}\bar{b}_{x_4}\bar{b}_{x_5}) \cdot (x_4 + x_5)]$. We can further simplify this expression by repeatedly using $[b_\mu + \bar{b}_\mu] = [1]$ for monomials μ to eliminate variables in B_X and $\bar{B}_X$ and obtain the equivalent expression $[\bar{b}_{x_1}\bar{b}_{x_1^2}(x_4 + x_5)]$. Since $[\bar{b}_\nu] = [\bar{b}_\nu][\bar{b}_\mu]$ whenever $\mu = \nu \cdot \nu'$ for some monomials ν, ν' and μ, we may conclude that

$$([x_4 + x_5] - [x_1]) - [2x_1^2 + x_1(x_4 + x_5)] = [\bar{b}_{x_1}(x_4 + x_5)].$$

Note that one can always replace elements from B_X and $\bar{B}_X$ by using the recipe that $\bar{b}_\mu$ is equivalent to $(1-\mu)$ and b_μ is equivalent to $(1-(1-\mu))$. In other words, $[\bar{b}_{x_1}(x_4+x_5)]$ can also be interpreted as $[x_4 + x_5] - [x_1]$. Although the latter expression does not precisely represent the how-provenance of μ_{27}, it provides a more succinct, albeit equivalent, representation of the provenance of μ_{27}.

A second use of expanded versions consists of finding a normal form of expressions in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$.

Example 7.7. When we have determined the expanded version

$$p_e[X_B] = \sum_{(P,N) \in S_p\text{-supp}(p[X_B])} \mathbf{b}_{P,N} \cdot r_{P,N}[X]$$

of $p[X_B]$, we can obtain an expression without $\mathbf{b}_{P,N} = \prod_{\mu \in P} b_\mu \cdot \prod_{\mu \in N} \bar{b}_\mu$ by replacing it by $\prod_{\mu \in P}(1 - (1 - \mu)) \cdot \prod_{\mu \in N}(1 - \mu)$. This expression in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ does not necessarily reflect the provenance but can be regarded as a normal form of elements in $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$. For instance, consider the annotation $[x_3] - [2x_1^2 + x_1(x_4 + x_5)]$ of μ_{28}. One can show that this expression is equivalent to $\bar{b}_{x_1}x_3 + b_{x_1}\bar{b}_{x_1^2}\bar{b}_{x_4}\bar{b}_{x_5}x_3$. By replacing $\bar{b}_\mu$ with $(1 - \mu)$ and b_μ with $(1 - (1 - \mu))$, the annotation of μ_{28} can be written equivalently as

$$(1 - x_1)x_3 + (1 - (1 - x_1))\big(1 - x_1^2\big)(1 - x_4)(1 - x_5)x_3.$$

Not surprisingly, moving to these normal forms may result in an exponential increase in the size of the annotations, similarly to the conversion of propositional logic expression into their disjunctive (or conjunctive) normal forms.

7.1. Factorization of SPARQL Query Evaluations

Let K and L be two spm-semirings. An spm-homomorphism $h : K \to L$ is said to *commute* with a SPARQL graph pattern P if, for any K-annotated RDF graph G_a,

$$[\![P]\!]_{h(G_a)} = h([\![P]\!]_{G_a}),$$

where $h(G_a)$ is the L-annotated RDF graph obtained from G_a by replacing $(s, p, o) \mapsto k$ by $(s, p, o) \mapsto h(k)$, for each K-annotated RDF triple $(s, p, o) \mapsto k$ in G_a, and $h([\![P]\!]_{G_a})$ is the L-annotated mapping set defined by $(h([\![P]\!]_{G_a}))(\mu) = h([\![P]\!]_{G_a}(\mu))$ for $\mu \in \mathcal{M}$.

A crucial ingredient for obtaining the factorization property is the following lemma.

LEMMA 7.8. *Let K and L be two spm-semirings. If $h : K \to L$ is an spm-semiring homomorphism, then h commutes with all SPARQL expressions.*

We omit the details of the proof because the lemma can easily be shown by induction on the structure of SPARQL graph patterns and by leveraging the properties of spm-semiring homomorphisms.

Let K be an spm-semiring and consider a valuation ν:$[X] \to K$, where $[X] = \{[x_1], \ldots, [x_n]\}$ is the set of equivalences corresponding to the variables in X in the universal spm-semiring. Denote by $Eval_\nu$ the unique homomorphism from $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ to K that coincides with ν on $[X]$. Recall that such a homomorphism exists by the universality of $(\mathbb{N}_\theta[X_B], +, \cdot, -, 0, 1)$ (cf. Proposition 6.25). It is an immediate consequence of Lemma 7.8 that the evaluation of SPARQL queries factors through the universal spm-semiring:

PROPOSITION 7.9. *Let K be an spm-semiring and P be a SPARQL graph pattern. For any K-annotated RDF graph G_a, we have that*

$$[\![P]\!]_{G_a} = Eval_\nu([\![P]\!]_{G/X}),$$

where G/X is the abstractly tagged version of G_a and $\nu : [X] \to K$ is the function that associates with each equivalence class $[x_i]$ the unique annotation of the triple in G_a tagged with $[x_i]$.

Proposition 7.9 says that one can deduce the right annotations for any spm-semiring K and any SPARQL expression P, given $[\![P]\!]_{G/X}$ and a valuation $\nu : [X] \to K$. We illustrate this for the bag, trust (set), and fuzzy settings.

Example 7.10. Consider the mapping μ_{18} in Figure 4(a). It has $[x_4 + x_5] - [x_1]$ as annotation in the universal spm-semiring. In the bag semantics of Example 2.1, we consider $(\mathbb{N}, +, \times, -_{bag}, 0, 1)$ as spm-semiring and let $\nu_b : [x_i] \to 1$ for all x_i. Then,

$$\begin{aligned} Eval_{\nu_b}([x_4 + x_5] - [x_1]) &= Eval_{\nu_b}([x_4 + x_5]) -_{bag} Eval_{\nu_b}([x_1]) \\ &= (Eval_{\nu_b}([x_4]) + Eval_{\nu_b}([x_5])) -_{bag} Eval_{\nu_b}([x_1]) \\ &= (\nu_b([x_4]) + \nu_b([x_5])) -_{bag} \nu_b([x_1]) \\ &= 2 -_{bag} 1 = 0. \end{aligned}$$

Similarly, for the trust semantics in Example 2.1, we consider $(\{\mathsf{true}, \mathsf{false}\}, \vee, \wedge, -_{trust}, \mathsf{true}, \mathsf{false})$ as spm-semiring and let $\nu_t : [x_i] \to \tau_i$ for all x_i. We obtain that

$$\begin{aligned} Eval_{\nu_b}([x_4 + x_5] - [x_1]) &= (\nu_t([x_4]) \vee \nu_t([x_5])) -_{trust} \nu_t([x_1]) \\ &= (\tau_4 \vee \tau_5) \wedge \bar{\tau}_1, \end{aligned}$$

as desired. The fuzzy semantics is verified similarly using the spm-semiring $([0, 1], \min, \max, -_{fuzzy}, 0, 1)$ given in Example 5.6.

8. SPARQL 1.1

So far, we considered the query language SPARQL 1.0 [Prud'hommeaux and Seaborne 2008]. As previously described, the SPARQL 1.0 algebra only uses the difference ($\setminus$) operator as an internal operator that is used to define the semantics of the operator `OPTIONAL`. In SPARQL 1.1 [Harris and Seaborne 2013], however, two non-monotone operators, `MINUS` and `NOT EXISTS`, are explicitly introduced.

The `MINUS` operator is closely related to the `DIFFERENCE` operator and can be defined in terms of an algebra operator on K-annotated mapping sets, as follows:

$$[\![P_1 \text{ MINUS } P_2]\!]_{G_a} := [\![P_1]\!]_{G_a} \setminus_{1.1} [\![P_2]\!]_{G_a}$$

where for K-annotated mapping sets Ω_1 and Ω_2 and mapping $\mu \in \mathcal{M}$:

$$(\Omega_1 \setminus_{1.1} \Omega_2)(\mu) := \Omega_1(\mu) \ominus \Big(\bigoplus_{\mu' \in \mathcal{M}, \mu \sim \mu', dom(\mu) \cap dom(\mu') \neq \emptyset} \Omega_2(\mu') \Big).$$

In other words, $\setminus_{1.1}$ only differs from $\setminus$ in that only compatible mappings are considered for which $dom(\mu) \cap dom(\mu') \neq \emptyset$ holds. Recall that $\setminus$ only required the compatibility

constraint. We claim that spm-semirings still provide the right annotation structure when dealing with $\setminus_{1.1}$. Indeed, to ensure that our framework carries over, we only need to verify whether we can enforce $\ominus$ to satisfy the identities of spm-semirings. More specifically, following the proof of Proposition 4.4, it suffices to verify that the following SPARQL equivalences hold:

$$\left\{ \begin{array}{rl} \Omega_1 \setminus_{1.1} \Omega_1 \equiv \emptyset & \Omega_1 \setminus_{1.1} (\Omega_2 \cup \Omega_3) \equiv (\Omega_1 \setminus_{1.1} \Omega_2) \setminus_{1.1} \Omega_3 \\ & \Omega_1 \bowtie (\Omega_2 \setminus_{1.1} \Omega_3) \equiv (\Omega_1 \bowtie \Omega_2) \setminus_{1.1} \Omega_3(cond) \\ (\Omega_1 \setminus_{1.1} (\Omega_1 \setminus_{1.1} \Omega_2)) \cup (\Omega_1 \setminus_{1.1} \Omega_2) & \equiv \Omega_1 \end{array} \right\}.$$

It is easily verified that all of these equivalences remain valid. We may thus conclude that one may add $\setminus_{1.1}$ to the K-annotated SPARQL algebra while still considering spm-semirings K.

Alternatively, it has been shown that $\setminus_{1.1.}$ can be expressed in terms of $\overline{\bowtie}$, $\bowtie$ and a filter condition that tests the boundedness of variables [Angles and Gutierrez 2008]. In this way, one obtains provenance expressions for $\setminus_{1.1}$ expressed in terms of an equivalent SPARQL K-algebra expression without $\setminus_{1.1}$ but with $\setminus$. A similar approach has been put forward in Damásio et al. [2012]. Since we can forget about $\setminus_{1.1}$ in this approach and just work with the SPARQL K-algebra as defined in Section 3, spm-semirings still suffice as annotation structure.

A second form of negation in SPARQL 1.1 is introduced through the operator `NOT EXISTS` that is part of the `FILTER` construct. Intuitively, `FILTER NOT EXISTS` filters out certain mappings based on the nonexistence of compatible mappings in some subquery. To translate this operator in terms of K-annotated mapping sets, we generalize the $\setminus$ operator as follows: Let Ω_1 be a K-annotated mapping set and let $\mathbf{\Omega}_2 = \{\Omega_{2,1}, \ldots, \Omega_{2,k}\}$ be a collection of K-annotated mapping sets. Let $\omega : \Omega_1 \to \mathbf{\Omega}_2$ be a function that assigns to each $\mu \in \mathcal{M}$ such that $\Omega_1(\mu) \neq 0$, a mapping set in $\mathbf{\Omega}_2$. We define the following operator:

$$(\Omega_1 \mapsto_\omega \mathbf{\Omega}_2)(\mu) := \bigoplus_{\mu' \in \mathcal{M}, \mu \sim \mu'} \omega(\mu)(\mu').$$

In other words, this operator extracts for each element μ in the support of Ω_1 the compatible mappings in the mapping set $\omega(\mu) \in \mathbf{\Omega}_2$. We generalize the difference operator such that, for a given mapping μ in Ω_1, only mappings $\Omega_1 \mapsto_\omega \mathbf{\Omega}_2$ are subtracted. More specifically,

$$\Omega_1 \setminus_\omega \mathbf{\Omega}_2(\mu) := \Omega_1(\mu) \ominus (\Omega_1 \mapsto_\omega \mathbf{\Omega}_2)(\mu).$$

Note that $\setminus_\omega$ indeed generalizes $\setminus$. To see this, it suffices to let $\mathbf{\Omega}_2$ consist of a single mapping set Ω_2 and let $\omega(\mu) = \Omega_2$ for each μ in Ω_1. It is easily verified that, for these choices, $\Omega_1 \setminus_\omega \mathbf{\Omega}_2$ coincides with $\Omega_1 \setminus \Omega_2$.

We next define the operator `FILTER NOT EXISTS` in terms of the $\setminus_\omega$ operation on K-annotated mapping sets. Given two graph patterns P_1 and P_2 and a K-annotated mapping set $[\![P_1]\!]_{G_a}$ we consider a set of mapping sets

$$[\![\mathbf{P_2}]\!]_{G_a,[P_1]_{G_a}} = \{[\![\mu(P_2)]\!]_{G_a} \mid \mu \in [\![P_1]\!]_{G_a}\}$$

that will play the role of $\mathbf{\Omega}_2$, and define $\omega(\mu) = [\![\mu(P_2)]\!]_{G_a} \in [\![\mathbf{P_2}]\!]_{G_a,[P_1]_{G_a}}$. Here, $\mu(P_2)$ is the graph pattern formed by replacing every occurrence of a variable $?x$ in P_2 by $\mu(?x)$, for each $?x \in dom(\mu)$. Finally, we define

$$[\![P_1 \text{ FILTER NOT EXISTS } P_2]\!]_{G_a} := [\![P_1]\!]_{G_a} \setminus_\omega [\![\mathbf{P_2}]\!]_{G_a,[P_1]_{G_a}}.$$

To our knowledge, `FILTER NOT EXISTS` has only been given a set semantics interpretation in Harris and Seaborne [2013]. When K is $(\{\mathsf{true}, \mathsf{false}\}, \vee, \wedge, -_{trust}, \mathsf{false}, \mathsf{true})$, one

can easily verify that the previous definition returns

$$\{\mu \in [\![P_1]\!]_{G_a} \mid \text{not exists a compatible } \mu' \in [\![\mu(P_2)]\!]_{G_a}\},$$

as given by the standard set semantics.

We saw earlier that $\setminus_\omega$ generalizes $\setminus$ when $\mathbf{\Omega}_2$ is restricted to be a single mapping set Ω_2 and ω maps every mapping in Ω_1 to Ω_2. Hence, following the same argument as for `MINUS`, we can infer that the $\ominus$ operator still satisfies the identities that define spm-semirings. We may thus conclude that $\setminus_\omega$ can be added to the K-annotated SPARQL algebra while still considering spm-semirings K.

9. RELATED WORK

The work presented in this article is inspired by the algebraic approach to modeling data provenance initiated by Green et al. [2007]. In that work, various forms of annotated relational data and their transformations by means of positive relational queries are considered. It is shown that the standard positive relational algebra equivalences hold on annotated relational data if and only if the annotations have the structure of a (commutative) semiring [Green et al. 2007]. Furthermore, they propose the semiring of *polynomials*—the *universal* semiring—as a provenance model that generalizes many forms of annotations and previously proposed provenance models [Green et al. 2007; Green 2011]. Furthermore, due to its universality, all annotation computations are shown to factor through the semiring of polynomials. This work has been extended to the semistructured setting in which transformations are expressed in a subset of XQuery [Foster et al. 2008], for which the authors showed that semirings are still an appropriate annotation structure. Similarly, semirings are sufficient for *positive* SPARQL queries on annotated RDF data, as described in Proposition 4.2 (Section 4) and as previously observed in Theoharis et al. [2011] and in Dividino et al. [2009] for idempotent semirings. We showed in this article that all key results of Green et al. [2007] carry over when spm-semirings are considered in the context of RDF and SPARQL.

The situation becomes more challenging when non-monotone query operators are taken into account [Geerts and Poggi 2010; Amsterdamer et al. 2011a, 2011b; Theoharis et al. 2011; Glavic and Alonso 2009]. In the relational setting, Geerts and Poggi [2010] extended semirings with a *monus* operation that captures the semantics of relational difference and proposed a universal *monus*-semiring, or m-semiring for short, as a provenance model. However, this universal m-semiring does not allow for a simple representation of its elements. Indeed, it is built up from formal terms that require the arbitrary nesting of expressions of the form $p[X] -_m q[X]$, where $p[X]$ and $q[X]$ are terms that are built up from variables in X, $+$, $\times$ and the monus $-_m$. By contrast, elements in an spm-semiring do have a normal form (using the expanded version) as shown in Example 7.7. Note, however, that these normal forms can be exponential in the size of the number of triples in G, and the structure of the SPARQL query gets lost in the normal form.

A study of the properties of m-semirings can be found in Amsterdamer et al. [2011a]. More specifically, a set of identities $\mathcal{E}_m$ is identified that characterizes m-semirings. Since the relational difference satisfies two sets of incompatible equivalences, $\mathcal{E}_s$ and $\mathcal{E}_b$, in the set and bag semantics, respectively, $\mathcal{E}_m$ only considers the common identities in these sets. As a consequence, certain intuitive equivalences of the relational algebra, such as $R \bowtie (S \setminus T) = (R \bowtie S) \setminus (R \bowtie T)$, are not necessarily satisfied on m-semiring annotated relational data [Amsterdamer et al. 2011a].

The classes of m-semirings and spm-semirings are incomparable. Indeed, there are identities that hold for m-semirings but not for spm-semirings and vice versa. For example, spm-semirings satisfy $\mathsf{id}_{11} : k_1 \otimes (k_2 \ominus k_3) = (k_1 \otimes k_2) \ominus k_3$, whereas m-semirings

satisfy $k_1 \otimes (k_2 \ominus k_3) = (k_1 \otimes k_2) \ominus (k_1 \otimes k_3)$ (identity a_{13} in Amsterdamer et al. [2011a]). One can verify that id_{11} implies a_{13} but not vice versa. Conversely, $k_1 \oplus (k_2 \ominus k_1) = k_2 \oplus (k_1 \ominus k_2)$ (identity a_{11} in Amsterdamer et al. [2011a]) holds for *m*-semirings but is not satisfied by spm-semirings.

Amsterdamer et al. [2011b] obtained an alternative semantics for relational difference based on their semantics for queries with *aggregation* on annotated relations through an encoding of difference using aggregation. Interestingly, the semantics of the difference defined in this manner is similar to the semantics of SPARQL difference. However, the resulting annotations reflect the encoding of difference through aggregation and thus do not provide a very intuitive description of the actual operations in the original query. By contrast, when spm-semiring annotations are just propagated through the query operations, they do provide a compact and actual representation of the structure of the query (see the discussion in Section 7). Furthermore, Amsterdamer et al. [2011b] do not propose a universal object that could be used as the provenance model for queries with difference under these semantics. Similarly to SPARQL difference, aggregation-based difference fails to satisfy a_{11} that holds for *m*-semirings. However, it satisfies a_{13}, which is not satisfied by all spm-semirings. This implies that even if semirings equipped with the aggregate-based difference are spm-semirings, the resulting class of algebraic structures must necessarily be a strict subset of spm-semirings.

The Perm system [Glavic and Alonso 2009] employs a provenance model that captures relational difference and outer join under set and bag semantics. However, it cannot capture SPARQL `DIFFERENCE` and `OPTIONAL` directly since their semantics differs from that of the corresponding relational operators, as explained in Section 2. Moreover, the provenance model of Perm, which is akin to why-provenance extended with $\wedge\neg$ for difference, is less informative than spm-semirings and does not suffice for computing annotations such as ranked trust or multiplicities for bag semantics.

In the semantic Web community, work has also been done on using algebraic structures to model and unify annotations for RDFS reasoning and SPARQL query answering [Damásio et al. 2012; Dividino et al. 2009; Zimmermann et al. 2012; Straccia 2013; Udrea et al. 2010; Buneman and Kostylev 2011].

Damásio et al. [2012] employ *m*-semirings to capture the semantics of SPARQL query answering over annotated RDF. More precisely, they use (m, δ)-semirings, which are *m*-semirings extended with a duplicate elimination operator δ, as introduced in Geerts and Poggi [2010]. Then, they encode SPARQL difference through a complex relational expression involving joins, relational set difference, and duplicate elimination. However, (m, δ)-semirings have the same deficiency as *m*-semirings: Their universal structure does not allow for a simple representation of its elements and is completely symbolic and not amenable to algebraic manipulation. For this reason, it is not as well-suited to be used as a provenance model as the structure we propose in Section 6. Indeed, in order to use the resulting expressions to compute (e.g., trust annotations), Damásio et al. [2012] resort to a simpler model by fixing the duplicate elimination function δ, thereby disregarding all (m, δ)-semirings with a more complex δ. Furthermore, similar to the approach taken in Amsterdamer et al. [2011b], the resulting expressions do not reflect the structure of operators in the original SPARQL query. As previously mentioned, when the universal spm-semiring is used as provenance structure without involving the expanded versions in the corresponding seba-structure, we obtain a compact and intuitive recording of the query's provenance.

Dividino et al. [2009] study both reasoning and SPARQL query answering on RDF$^+$ graphs. An RDF$^+$ graph is a *K*-annotated RDF graph where *K* is a boolean algebra. The corresponding, most general, provenance model is when *K* consists of boolean propositional logic formulas, up to equivalence. The presence of negation in *K* allows

Dividino et al. [2009] to generalize the semantics of OPTIONAL in terms of $\wedge\neg$, similar to our definition when trust and set semantics are concerned. We do not require $\oplus$ and $\otimes$ to be idempotent, and therefore they are not restricted to boolean algebras.

Zimmermann et al. [2012] also consider both reasoning and SPARQL query answering. They propose AnQL, an extension of SPARQL such that queries can also *explicitly* manipulate both data and annotations. In contrast, we only consider *implicit* provenance [Buneman et al. 2008], in which annotations are simply carried along when the data are queried using standard SPARQL queries. They provide a generalized semantics of SPARQL on K-annotated RDF graphs, where K is assumed to be an idempotent and $\top$-annihilating semiring. As previously mentioned, we do not require $\oplus$ and $\otimes$ to be idempotent. In fact, our work indicates that these restrictions are not required when SPARQL query answering is concerned. Furthermore, Zimmermann et al. [2012] do not follow an axiomatic approach based on query equivalences and do not identify a universal, most general, annotation domain either.

Udrea et al. [2010] consider reasoning and query answering on ARDF, an extension of RDF in which triples are annotated by a partially ordered set. They consider explicit manipulation of annotations and extend the semantics of a limited fragment of SPARQL that does not include the UNION, FILTER, and OPTIONAL (or DIFFERENCE) operators, in contrast to our work.

Finally, Buneman and Kostylev [2011] propose the use of idempotent and $\top$-annihilating semirings for generalizing the process of RDFS reasoning. Query answering is not considered in Buneman and Kostylev paper and is thus orthogonal to the focus of our work.

10. CONCLUSION AND FUTURE WORK

We presented spm-semirings, an extension of semirings to capture the semantics of SPARQL queries, involving the non-monotone operator OPTIONAL, on annotated RDF data. Moreover, we showed that spm-semirings have a universal structure that provides a concise representation of the provenance of RDF data and SPARQL queries with the OPTIONAL operator.

Furthermore, we showed how to construct spm-semirings by means of seba-structures by establishing that any spm-semiring can be derived from a seba-structure. We believe that this characterization is interesting in its own right.

Finally, we showed that, just as in the relational case [Green et al. 2007], provenance expressions from the universal spm-semiring can be recorded during query-answering and later be evaluated in appropriate spm-semirings in order to compute different forms of annotations for a variety of applications. In other words, we have unified the semantics of SPARQL on various RDF data models that are used in different application scenarios.

Some of these applications may not require the full expressiveness of the universal spm-semiring. For such applications, it may be desirable to record provenance expressions from a less informative model instead (e.g., if such expressions are more efficient to store and to evaluate than those of more informative provenance models). This raises the question of whether one can identify a clean hierarchy of provenance models, all residing within the class of spm-semirings, with the intent to capturing provenance at various levels of granularity. Furthermore, an interesting question is to formally study the relationship between spm-semirings, on the one hand, and the algebraic approaches [Damásio et al. 2012; Amsterdamer et al. 2011b], on the other hand. Finally, from a practical point of view, one would like to know the overhead caused by maintaining annotations in the universal spm-semiring during SPARQL query evaluation. We leave these issues as part of our future work.

APPENDIX

Proof of Proposition 4.6

We show that $\mathcal{A}' = \mathcal{A} \setminus \{\mathsf{id}_2, \mathsf{id}_3\}$ is minimal by providing for each id_i in $\mathcal{A}'$ an algebraic structure that satisfies $\mathcal{A}' \setminus \{\mathsf{id}_i\}$ but does not satisfy id_i. The structures here are obtained by the program Mace4 [McCune 2010] that searches for finite models and counterexamples for equational logic. They have been manually verified, however.

Consider the identity id_1: For all $k \in K$, $k \otimes 1 = k$. Consider $(K = \{0, 1\}, \oplus_1, \otimes_1, \ominus_1, 0, 1)$ with

$\oplus_1$	0	1
0	0	1
1	1	1

$\otimes_1$	0	1
0	0	0
1	0	0

$\ominus_1$	0	1
0	0	0
1	1	0

Clearly, $1 \otimes_1 1 = 0$ contradicts id_1. It is readily verified that $(K, \oplus_1, \otimes_1, \ominus_1, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_1\}$ by exhaustively verifying the identities in $\mathcal{A}' \setminus \{\mathsf{id}_1\}$.

Consider the identity id_4: For all $k_1, k_2 \in K$, $k_1 \oplus k_2 = k_2 \oplus k_1$. Consider $(K = \{0, 1, 2\}, \oplus_4, \otimes_4, \ominus_4, 0, 1)$ with

$\oplus_4$	0	1	2
0	0	1	2
1	1	1	1
2	2	2	2

$\otimes_4$	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

$\ominus_4$	0	1	2
0	0	0	0
1	1	0	0
2	2	0	0

Clearly, $1 \oplus_4 2 = 1 \neq 2 \oplus_4 1 = 2$ contradicts id_4. It is readily verified that $(K, \oplus_4, \otimes_4, \ominus_4, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_4\}$.

Consider the identity id_5: For all $k_1, k_2 \in K$, $k_1 \otimes k_2 = k_2 \otimes k_1$. Consider $(K = \{0, 1, 2\}, \oplus_5, \otimes_5, \ominus_5, 0, 1)$ with

$\oplus_5$	0	1	2
0	0	1	2
1	1	1	2
2	2	2	2

$\otimes_5$	0	1	2
0	0	0	2
1	0	1	2
2	0	2	2

$\ominus_5$	0	1	2
0	0	0	0
1	1	0	0
2	2	0	0

Clearly, $0 \otimes_5 2 = 2 \neq 2 \otimes_5 0 = 0$ contradicts id_5. It is readily verified that $(K, \oplus_5, \otimes_5, \ominus_5, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_5\}$.

Consider the identity id_6: For all $k_1, k_2, k_3 \in K$, $k_1 \oplus (k_2 \oplus k_3) = (k_1 \oplus k_2) \oplus k_3$. Consider $(K = \{0, 1, 2, 3\}, \oplus_6, \otimes_6, \ominus_6, 0, 1)$ with

$\oplus_6$	0	1	2	3
0	0	1	2	3
1	1	1	3	1
2	2	3	2	3
3	3	1	3	3

$\otimes_6$	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	3

$\ominus_6$	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	2	0	0	0
3	3	0	0	0

Clearly, $1 \oplus_6 (1 \oplus_6 2) = 1 \neq (1 \oplus_6 1) \oplus_6 2 = 3$ contradicts id_6. It is readily verified that $(K, \oplus_6, \otimes_6, \ominus_6, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_6\}$.

Consider the identity id_7: For all $k_1, k_2, k_3 \in K$, $k_1 \otimes (k_2 \otimes k_3) = (k_1 \otimes k_2) \otimes k_3$. Consider $(K = \{0, 1, 2, 3\}, \oplus_7, \otimes_7, \ominus_7, 0, 1)$ with

$\oplus_7$	0	1	2	3
0	0	1	2	3
1	1	1	1	3
2	2	1	2	3
3	3	3	3	3

$\otimes_7$	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	1
3	0	3	1	3

$\ominus_7$	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	2	0	0	0
3	3	0	0	0

Clearly, $2 \otimes_7 (2 \otimes_7 3) = 2 \neq (2 \otimes_7 2) \otimes_7 3 = 0$ contradicts id_7. It is readily verified that $(K, \oplus_7, \otimes_7, \ominus_7, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_7\}$.

Consider the identity id_8: For all $k_1, k_2, k_3 \in K$, $k_1 \otimes (k_2 \oplus k_3) = (k_1 \otimes k_2) \oplus (k_1 \otimes k_3)$. Consider $(K = \{0, 1\}, \oplus_8, \otimes_8, \ominus_8, 0, 1)$ with

$\oplus_8$	0	1
0	0	1
1	1	0

$\otimes_8$	0	1
0	1	0
1	0	1

$\ominus_8$	0	1
0	0	1
1	1	0

Clearly, $0 \otimes_8 (0 \oplus_8 0) = 1 \neq (0 \otimes_8 0) \oplus_8 (0 \otimes_8 0) = 0$ contradicts id_8. It is readily verified that $(K, \oplus_8, \otimes_8, \ominus_8, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_8\}$.

Consider the identity id_9: For all $k_1 \in K$, $k_1 \ominus k_1 = 0$. Consider $(K = \{0, 1\}, \oplus_9, \otimes_9, \ominus_9, 0, 1)$ with

$\oplus_9$	0	1
0	0	0
1	0	1

$\otimes_9$	0	1
0	0	0
1	0	1

$\ominus_9$	0	1
0	0	0
1	1	1

Clearly, $1 \ominus_9 1 = 1$ contradicts id_9. It is readily verified that $(K, \oplus_9, \otimes_9, \ominus_9, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_9\}$.

Consider the identity id_{10}: For all $k_1, k_2, k_3 \in K$, $k_1 \ominus (k_2 \oplus k_3) = (k_1 \ominus k_2) \ominus k_3$. Consider $(K = \{0, 1\}, \oplus_{10}, \otimes_{10}, \ominus_{10}, 0, 1)$ with

$\oplus_{10}$	0	1
0	0	1
1	1	0

$\otimes_{10}$	0	1
0	0	0
1	0	1

$\ominus_{10}$	0	1
0	0	0
1	1	0

Clearly, $1 \ominus_{10} (1 \oplus_{10} 1) = 1 \neq (1 \ominus_{10} 1) \ominus_{10} 1 = 0$ contradicts id_{10}. It is readily verified that $(K, \oplus_{10}, \otimes_{10}, \ominus_{10}, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_{10}\}$.

Consider the identity id_{11}: For all $k_1 \in K$, $k_1 \otimes (k_2 \ominus k_3) = (k_1 \otimes k_2) \ominus k_3$. Consider $(K = \{0, 1\}, \oplus_{11}, \otimes_{11}, \ominus_{11}, 0, 1)$ with

$\oplus_{11}$	0	1
0	0	1
1	1	0

$\otimes_{11}$	0	1
0	0	0
1	0	1

$\ominus_{11}$	0	1
0	0	1
1	1	0

Clearly, $0 \otimes_{11} (0 \ominus_{11} 1) = 0 \neq (0 \otimes_{11} 0) \ominus_{11} 1 = 1$ contradicts id_{11}. It is readily verified that $(K, \oplus_{11}, \otimes_{11}, \ominus_{11}, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_{11}\}$.

Finally, consider the identity id_{12}: For all $k_1, k_2 \in K$, $(k_1 \ominus (k_1 \ominus k_2)) \oplus (k_1 \ominus k_2) = k_1$. Consider $(K = \{0, 1\}, \oplus_{12}, \otimes_{12}, \ominus_{12}, 0, 1)$ with

$\oplus_{12}$	0	1
0	0	0
1	0	0

$\otimes_{12}$	0	1
0	0	0
1	0	1

$\ominus_{12}$	0	1
0	0	0
1	0	0

Clearly, $(1 \ominus_{12} (1 \ominus_{12} 0)) \oplus_{12} (1 \ominus_{12} 0) = 0 \neq 1$ contradicts id_{12}. It is readily verified that $(K, \oplus_{12}, \otimes_{12}, \ominus_{12}, 0, 1) \models \mathcal{A}' \setminus \{\mathsf{id}_{12}\}$.

Proof of Theorem 5.3

We show that every spm-semiring is derived from some seba-structure, and, vice versa, every structure derived from a seba-structure is an spm-semiring. We start by showing that, given a seba-structure $(K, B, d, \imath)$, its derived structure $(K, \oplus, \otimes, \ominus, 0, 1)$ is an spm-semiring. We do this by verifying the identities $\mathcal{A}'$ given in Figure 2. Since $(K, \oplus, \otimes, 0, 1)$ is a semiring, the identities id_1–id_8 are already satisfied. It thus remains to show that id_9–id_{12} hold as well. We do this by leveraging the fact that K is a semiring, B a boolean algebra, and the properties of the mappings d and $\imath$ as stated in the definition of seba-structures (Definition 5.1).

Let $k_1, k_2, k_3 \in K$. We first show that id_9 holds:

$$\begin{aligned} k_1 \ominus k_1 &= k_1 \otimes \imath(\overline{d(k_1)}) && \text{(by def.)} \\ &= 0. && (\text{by } \mathsf{sb}_7) \end{aligned}$$

Also, id_{10} holds:

$$\begin{aligned} k_1 \ominus (k_2 \oplus k_3) &= k_1 \otimes \imath(\overline{d(k_2 \oplus k_3)}) && \text{(by def.)} \\ &= k_1 \otimes \imath(\overline{d(k_2) \vee d(k_3)}) && (\text{by } \mathsf{sb}_3) \\ &= k_1 \otimes \imath(\overline{d(k_2)} \wedge \overline{d(k_3)}) && \text{(De Morgan)} \\ &= k_1 \otimes \left(\imath(\overline{d(k_2)}) \otimes \imath(\overline{d(k_3)})\right) && (\text{by } \mathsf{sb}_6) \\ &= \left(k_1 \otimes \imath(\overline{d(k_2)})\right) \otimes \imath(\overline{d(k_3)}) && (\text{by } \mathsf{id}_7) \\ &= (k_1 \ominus k_2) \ominus k_3. && \text{(by def.)} \end{aligned}$$

Similarly, id_{11} holds:

$$\begin{aligned} k_1 \otimes (k_2 \ominus k_3) &= k_1 \otimes (k_2 \otimes \imath(\overline{d(k_3)})) && \text{(by def.)} \\ &= (k_1 \otimes k_2) \otimes \imath(\overline{d(k_3)}) && (\text{by } \mathsf{id}_7) \\ &= (k_1 \otimes k_2) \ominus k_3. && \text{(by def.)} \end{aligned}$$

Finally, we verify that id_{12} holds:

$$\begin{aligned} (k_1 \ominus (k_1 \ominus k_2)) \oplus (k_1 \ominus k_2) &= \left(k_1 \otimes \imath(\overline{d(k_1 \otimes \imath(\overline{d(k_2)}))})\right) \oplus \left(k_1 \otimes \imath(\overline{d(k_2)})\right) && \text{(by def.)} \\ &= \left(k_1 \otimes \imath(\overline{d(k_1) \wedge \overline{d(k_2)}})\right) \oplus \left(k_1 \otimes \imath(\overline{d(k_2)})\right) && (\text{by } \mathsf{sb}_5) \\ &= \left(k_1 \otimes \imath(\overline{d(k_1)} \vee d(k_2))\right) \oplus \left(k_1 \otimes \imath(\overline{d(k_2)})\right) && \text{(De Morgan)} \\ &= \left(k_1 \otimes (\imath(\overline{d(k_1)}) \oplus (\imath(d(k_1)) \otimes \imath(d(k_2))))\right) \oplus (k_1 \otimes \imath(\overline{d(k_2)})) && (\text{by } \mathsf{sb}_4) \\ &= \left(k_1 \otimes \imath(\overline{d(k_1)})\right) \oplus \left(k_1 \otimes (\imath(d(k_1)) \otimes \imath(d(k_2)))\right) \oplus \left(k_1 \otimes \imath(\overline{d(k_2)})\right) && (\text{by } \mathsf{id}_8) \end{aligned}$$

$$
\begin{aligned}
&= \big((k_1 \otimes \iota(d(k_1))) \otimes \iota(d(k_2))\big) \oplus \big(k_1 \otimes \iota(\overline{d(k_2)})\big) && \text{(by } \mathsf{sb}_7, \mathsf{id}_3, \mathsf{id}_7\text{)}\\
&= \big(k_1 \otimes \iota(d(k_2))\big) \oplus \big(k_1 \otimes \iota(\overline{d(k_2)})\big) && \text{(by } k_1 \otimes \iota(d(k_1)) = k_1\text{)}\\
&= k_1 \otimes \big(\iota(d(k_2)) \oplus \iota(\overline{d(k_2)})\big) && \text{(by } \mathsf{id}_8\text{)}\\
&= k_1 \otimes \big(\iota(d(k_2)) \oplus (\iota(\overline{d(k_2)}) \otimes 1)\big) && \text{(by } \mathsf{id}_1\text{)}\\
&= k_1 \otimes \big(\iota(d(k_2)) \oplus (\iota(\overline{d(k_2)}) \otimes \iota(\top))\big) && \text{(by } \mathsf{sb}_2\text{)}\\
&= k_1 \otimes \big(\iota(d(k_2) \vee \top)\big) && \text{(by } \mathsf{sb}_4\text{)}\\
&= k_1 \otimes (\iota(\top)) && \text{(by } b \vee \top = \top\text{)}\\
&= k_1. && \text{(by } \mathsf{id}_1, \mathsf{sb}_2\text{)}
\end{aligned}
$$

Observe that here we use known identities of boolean algebras and the identity that states that for any $k_1 \in K$, $k_1 \otimes \iota(d(k_1)) = k_1$. The latter identity holds in every seba-structure:

$$
\begin{aligned}
k_1 &= k_1 \otimes 1 && \text{(by } \mathsf{id}_1\text{)}\\
&= k_1 \otimes \iota(\top) && \text{(by } \mathsf{sb}_2\text{)}\\
&= k_1 \otimes \iota(d(k_1) \vee \overline{d(k_1)}) && \text{(by } b \vee \overline{b} = \top\text{)}\\
&= k_1 \otimes \big(\iota(d(k_1)) \oplus (\iota(\overline{d(k_1)}) \otimes \iota(\overline{d(k_1)}))\big) && \text{(by } \mathsf{sb}_4\text{)}\\
&= \big(k_1 \otimes (\iota(d(k_1)))\big) \oplus \big(k_1 \otimes (\iota(\overline{d(k_1)}) \otimes \iota(\overline{d(k_1)}))\big) && \text{(by } \mathsf{id}_8\text{)}\\
&= \big(k_1 \otimes (\iota(d(k_1)))\big) \oplus \big((k_1 \otimes \iota(\overline{d(k_1)})) \otimes \iota(\overline{d(k_1)})\big) && \text{(by } \mathsf{id}_7\text{)}\\
&= k_1 \otimes (\iota(d(k_1))). && \text{(by } \mathsf{id}_2, \mathsf{id}_3, \mathsf{sb}_7\text{)}
\end{aligned}
$$

From this, we may conclude that an algebraic structure derived from a seba-structure is indeed an spm-semiring.

For the converse, we need to show that every spm-semiring is derived from some seba-structure. Let $(K, \oplus, \otimes, \ominus, 0, 1)$ be an spm-semiring. We next define a boolean algebra $(B, \vee, \wedge, \bot, \top)$ together with the mappings $d : K \to B$ and $\iota : B \to K$, such that $(K, \oplus, \otimes, \ominus, 0, 1)$ is derived from (K, B, d, ι).

Let $d : K \to K$ be defined such that for every $k \in K$,

$$d(k) := 1 \ominus (1 \ominus k).$$

Furthermore, the domain of the boolean algebra is defined as

$$B := \{b \in K \mid \exists k \in K, b = d(k)\}.$$

In other words, d is a mapping from K to B. We equip B with disjunction ($\vee$), conjunction ($\wedge$), and complementation ($\bar{}$), as follows. Let b, b_1, and b_2 be elements in B. In addition, let k, k_1, and k_2 be elements in K such that $b = d(k)$, $b_1 = d(k_1)$, and $b_2 = d(k_2)$. We define

$$b_1 \vee b_2 := 1 \ominus (1 \ominus (k_1 \oplus k_2)), \quad \bar{b} := 1 \ominus k \quad \text{and} \quad b_1 \wedge b_2 := \overline{\overline{b_1} \vee \overline{b_2}}.$$

Finally, we set $\bot := d(0)$ and $\top := d(1)$ and define $\iota : B \to K$ to be the identity mapping.

We conclude the proof of Theorem 5.3 by showing the following three claims:

Claim 1. *The operations $\vee$, $\wedge$, $\bar{}$ on B are well-defined. Furthermore, the structure $(B, \vee, \wedge, \bar{}, \bot, \top)$ is a boolean algebra.*

Claim 2. *(K, B, d, ι) is a seba-structure.*

Claim 3. *$(K, \oplus, \otimes, \ominus, 0, 1)$ is derived from (K, B, d, ι).*

id_1: $k \otimes 1 = k$
id_2: $k \otimes 0 = 0$
id_3: $k \oplus 0 = k$
id_4: $k_1 \oplus k_2 = k_2 \oplus k_1$
id_5: $k_1 \otimes k_2 = k_2 \otimes k_1$
id_6: $k_1 \oplus (k_2 \oplus k_3) = (k_1 \oplus k_2) \oplus k_3$
id_7: $k_1 \otimes (k_2 \otimes k_3) = (k_1 \otimes k_2) \otimes k_3$
id_8: $k_1 \otimes (k_2 \oplus k_3) = (k_1 \otimes k_2) \oplus (k_1 \otimes k_3)$
id_9: $k \ominus k = 0$
id_{10}: $k_1 \ominus (k_2 \oplus k_3) = (k_1 \ominus k_2) \ominus k_3$
id_{11}: $k_1 \otimes (k_2 \ominus k_3) = (k_1 \otimes k_2) \ominus k_3$
id_{12}: $\big(k_1 \ominus (k_1 \ominus k_2)\big) \oplus (k_1 \ominus k_2) = k_1$

id_{13}: $k_1 \ominus 0 = k_1$
id_{14}: $(k_1 \oplus k_2) \ominus k_3 = (k_1 \ominus k_3) \oplus (k_2 \ominus k_3)$
id_{15}: $(k_1 \ominus k_2) \otimes (k_3 \ominus k_4) = (k_1 \otimes k_3) \ominus (k_2 \oplus k_4)$
id_{16}: $\big(k_1 \ominus (k_1 \ominus (k_1 \ominus k_2))\big) = k_1 \ominus k_2$
id_{17}: $(k_1 \ominus k_2) \ominus (1 \ominus k_3) = k_1 \ominus (1 \ominus (k_3 \ominus k_2))$
id_{18}: $k_1 \ominus (k_2 \oplus k_2) = k_1 \ominus k_2$

Fig. 5. Identities used in the proof of Claims 1, 2, and 3.

It remains to verify the three claims, which we do next.

We start by verifying that a number of identities, id_{13}–id_{18}, as shown in Figure 5, are implied by the identities $\mathcal{A}'$ that define spm-semirings. These auxilliary identities will be used extensively in the proofs of the three claims. Recall that, apart from the identities in $\mathcal{A}'$, we may also use id_2 and id_3 because these are already shown to be implied by $\mathcal{A}'$ in Proposition 4.6. We further observe that id_{13} is shown to be implied by $\mathcal{A}'$ in Proposition 4.4.

The following derivations have been both manually verified and verified with Prover9 [McCune 2010], which proves identities in equational logic, among other things.

We start by showing that $\mathcal{A}' \models \mathsf{id}_{14}$. This identity states that $(k_1 \oplus k_2) \ominus k_3 = (k_1 \ominus k_3) \oplus (k_2 \ominus k_3)$ for all k_1, k_2 and k_3 in K.

$$\begin{aligned}
(k_1 \oplus k_2) \ominus k_3 &= (k_1 \oplus k_2) \otimes (1 \ominus k_3) && \text{(by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}\text{)} \\
&= (k_1 \otimes (1 \ominus k_3)) \oplus (k_2 \otimes (1 \ominus k_3)) && \text{(by } \mathsf{id}_8\text{)} \\
&= (k_1 \ominus k_3) \oplus (k_2 \ominus k_3). && \text{(by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}\text{)}
\end{aligned}$$

We also have that $\mathcal{A}' \models \mathsf{id}_{15}$, where id_{15} states that for all $k_1, k_2, k_3,$ and k_4 in K, $(k_1 \ominus k_2) \otimes (k_3 \ominus k_4) = (k_1 \otimes k_3) \ominus (k_2 \oplus k_4)$.

$$\begin{aligned}
(k_1 \ominus k_2) \otimes (k_3 \ominus k_4) &= (k_1 \otimes (1 \ominus k_2)) \otimes (k_3 \otimes (1 \ominus k_4)) && \text{(by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}\text{)} \\
&= (k_1 \otimes k_3) \otimes (1 \ominus k_2) \otimes (1 \ominus k_4) && \text{(by } \mathsf{id}_5 \text{ and } \mathsf{id}_7\text{)} \\
&= (k_1 \otimes k_3) \otimes ((1 \ominus k_2) \ominus k_4) && \text{(by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}\text{)} \\
&= (k_1 \otimes k_3) \otimes (1 \ominus (k_2 \oplus k_4)) && \text{(by } \mathsf{id}_{10}\text{)} \\
&= (k_1 \otimes k_3) \ominus (k_2 \oplus k_4). && \text{(by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}\text{)}
\end{aligned}$$

We next verify that id_{16} is implied by $\mathcal{A}'$. This identity states that for all $k_1, k_2 \in K$, $k_1 \ominus (k_1 \ominus (k_1 \ominus k_2)) = k_1 \ominus k_2$. To show that $\mathcal{A}' \models \mathsf{id}_{16}$, we assume that the following additional identities are shown to be implied by $\mathcal{A}'$:

$$\mathsf{id}_{19}: \quad k_1 \ominus (k_1 \ominus (k_1 \ominus k_2)) = (k_1 \ominus k_2) \ominus (k_1 \ominus (k_1 \ominus k_2))$$
$$\mathsf{id}_{20}: \quad k_1 \ominus (k_1 \ominus (k_2 \ominus k_1)) = 0$$

Given these, we then have that $\mathcal{A}' \models \mathsf{id}_{16}$:

$$\begin{aligned}
k_1 \ominus k_2 &= (k_1 \ominus k_2) \ominus 0 && (\text{by } \mathsf{id}_{13})\\
&= (k_1 \ominus k_2) \ominus ((k_1 \ominus k_2) \ominus ((k_1 \ominus k_2) \ominus (k_1 \ominus (k_1 \ominus k_2)))) && (\text{by } \mathsf{id}_{20})\\
&= ((k_1 \ominus k_2) \ominus (k_1 \ominus (k_1 \ominus k_2)))\\
&\qquad \ominus ((k_1 \ominus k_2) \ominus ((k_1 \ominus k_2) \ominus (k_1 \ominus (k_1 \ominus k_2)))) && (\text{by } \mathsf{id}_{19})\\
&= (k_1 \ominus k_2) \ominus (k_1 \ominus (k_1 \ominus k_2)) && (\text{by } \mathsf{id}_{20})\\
&= k_1 \ominus (k_1 \ominus (k_1 \ominus k_2)). && (\text{by } \mathsf{id}_{19})
\end{aligned}$$

We next verify that $\mathcal{A}' \models \mathsf{id}_{19}$:

$$\begin{aligned}
k_1 \ominus (k_1 \ominus (k_1 \ominus k_2)) &= (k_1 \ominus (k_1 \ominus (k_1 \ominus k_2))) \oplus 0 && (\text{by } \mathsf{id}_{3})\\
&= (k_1 \ominus (k_1 \ominus (k_1 \ominus k_2)))\\
&\qquad \oplus ((k_1 \ominus (k_1 \ominus k_2)) \ominus (k_1 \ominus (k_1 \ominus k_2))) && (\text{by } \mathsf{id}_{9})\\
&= (k_1 \oplus (k_1 \ominus (k_1 \ominus k_2))) \ominus (k_1 \ominus (k_1 \ominus k_2)) && (\text{by } \mathsf{id}_{14})\\
&= (((k_1 \ominus k_2) \oplus (k_1 \ominus (k_1 \ominus k_2)))\\
&\qquad \oplus (k_1 \ominus (k_1 \ominus k_2))) \ominus (k_1 \ominus (k_1 \ominus k_2)) && (\text{by } \mathsf{id}_{4} \text{ and } \mathsf{id}_{12})\\
&= ((k_1 \ominus k_2) \ominus (k_1 \ominus (k_1 \ominus k_2))) \oplus 0 \oplus 0 && (\text{by } \mathsf{id}_{6}, \mathsf{id}_{9}, \text{ and } \mathsf{id}_{14})\\
&= (k_1 \ominus k_2) \ominus (k_1 \ominus (k_1 \ominus k_2)) && (\text{by } \mathsf{id}_{3})
\end{aligned}$$

To verify that $\mathcal{A}' \models \mathsf{id}_{20}$, we use the identity

$$\mathsf{id}_{21}: \quad (k_1 \otimes k_2) \ominus (k_1 \ominus (k_2 \ominus k_1)) = 0,$$

which is implied by $\mathcal{A}'$:

$$\begin{aligned}
0 &= k_2 \otimes 0 && (\text{by } \mathsf{id}_{2})\\
&= k_2 \otimes ((k_1 \ominus (k_2 \ominus k_1)) \ominus (k_1 \ominus (k_2 \ominus k_1))) && (\text{by } \mathsf{id}_{9})\\
&= ((k_2 \otimes k_1) \ominus (k_2 \ominus k_1)) \ominus (k_1 \ominus (k_2 \ominus k_1)) && (\text{by } \mathsf{id}_{11})\\
&= (0 \oplus ((k_2 \otimes k_1) \ominus (k_2 \ominus k_1))) \ominus (k_1 \ominus (k_2 \ominus k_1)) && (\text{by } \mathsf{id}_{3})\\
&= ((k_2 \otimes 0) \oplus ((k_2 \otimes k_1) \ominus (k_2 \ominus k_1))) \ominus (k_1 \ominus (k_2 \ominus k_1)) && (\text{by } \mathsf{id}_{2})\\
&= ((k_2 \otimes (k_1 \ominus k_1)) \oplus ((k_2 \otimes k_1) \ominus (k_2 \ominus k_1))) \ominus (k_1 \ominus (k_2 \ominus k_1)) && (\text{by } \mathsf{id}_{9})\\
&= ((k_1 \otimes (k_2 \ominus k_1)) \oplus (k_1 \otimes (k_2 \ominus (k_2 \ominus k_1)))) \ominus (k_1 \ominus (k_2 \ominus k_1)) && (\text{by } \mathsf{id}_{5} \text{ and } \mathsf{id}_{11})\\
&= (k_1 \otimes ((k_2 \ominus k_1) \oplus (k_2 \ominus (k_2 \ominus k_1)))) \ominus (k_1 \ominus (k_2 \ominus k_1)) && (\text{by } \mathsf{id}_{4} \text{ and } \mathsf{id}_{8})\\
&= (k_1 \otimes k_2) \ominus (k_1 \ominus (k_2 \ominus k_1)). && (\text{by } \mathsf{id}_{12})
\end{aligned}$$

From this, it follows that $\mathcal{A}' \models \mathsf{id}_{20}$:

$$\begin{aligned}
k_1 \ominus (k_1 \ominus (k_2 \ominus k_1)) &= (k_1 \ominus (k_1 \ominus (k_2 \ominus k_1))) \oplus 0 && (\text{by } \mathsf{id}_{3})\\
&= (k_1 \ominus (k_1 \ominus (k_2 \ominus k_1))) \oplus ((k_1 \otimes k_2) \ominus (k_1 \ominus (k_2 \ominus k_1))) && (\text{by } \mathsf{id}_{21})\\
&= (k_1 \oplus (k_1 \otimes k_2)) \ominus (k_1 \ominus (k_2 \ominus k_1)) && (\text{by } \mathsf{id}_{14})\\
&= k_1 \otimes ((1 \oplus k_2) \ominus (k_1 \ominus (k_2 \ominus k_1))) && (\text{by } \mathsf{id}_{1}, \mathsf{id}_{8}, \text{ and } \mathsf{id}_{11})\\
&= k_1 \otimes ((1 \oplus k_2) \ominus ((k_1 \otimes 1) \ominus (k_2 \ominus k_1))) && (\text{by } \mathsf{id}_{1})\\
&= k_1 \otimes ((1 \oplus k_2) \ominus ((k_1 \otimes ((1 \ominus k_1) \oplus (1 \ominus (1 \ominus k_1)))) \ominus (k_2 \ominus k_1))) && (\text{by } \mathsf{id}_{12})\\
&= k_1 \otimes ((1 \oplus k_2) \ominus ((k_1 \ominus k_1) \oplus (k_1 \ominus (1 \ominus k_1))) \ominus (k_2 \ominus k_1)) && (\text{by } \mathsf{id}_{8} \text{ and } \mathsf{id}_{11})
\end{aligned}$$

$$
\begin{aligned}
&= k_1 \otimes ((1 \oplus k_2) \ominus (0 \oplus (k_1 \ominus (1 \ominus k_1))) \ominus (k_2 \ominus k_1)) && \text{(by } \mathsf{id}_9\text{)}\\
&= k_1 \otimes ((1 \oplus k_2) \ominus ((k_1 \ominus (1 \ominus k_1)) \ominus (k_2 \ominus k_1))) && \text{(by } \mathsf{id}_3\text{)}\\
&= k_1 \otimes ((1 \oplus k_2) \ominus (k_1 \ominus ((1 \ominus k_1) \oplus (k_2 \ominus k_1)))) && \text{(by } \mathsf{id}_{10}\text{)}\\
&= k_1 \otimes ((1 \oplus k_2) \ominus (k_1 \ominus ((1 \oplus k_2) \ominus k_1))) && \text{(by } \mathsf{id}_{14}\text{)}\\
&= (k_1 \otimes (1 \oplus k_2)) \ominus (k_1 \ominus ((1 \oplus k_2) \ominus k_1)) && \text{(by } \mathsf{id}_{11}\text{)}\\
&= 0. && \text{(by } \mathsf{id}_{21}\text{)}
\end{aligned}
$$

We next consider $\mathsf{id}_{17,}$ which states that $(k_1 \ominus k_2) \ominus (1 \ominus k_3) = k_1 \ominus (1 \ominus (k_3 \ominus k_2))$ for all k_1, k_2 and k_3 in K. To verify that $\mathcal{A}' \models \mathsf{id}_{17,}$ we need the following auxiliary identities:

$$
\begin{aligned}
&\mathsf{id}_{22}\text{:} \quad k_1 \ominus (1 \ominus k_2) = k_1 \ominus (k_1 \ominus k_2)\\
&\mathsf{id}_{23}\text{:} \quad (k_1 \ominus k_2) \ominus (k_3 \ominus k_2) = (k_1 \ominus k_2) \ominus k_3\\
&\mathsf{id}_{24}\text{:} \quad k_1 \ominus (1 \ominus (k_3 \ominus k_2)) = (k_1 \ominus k_2) \ominus (1 \ominus (k_3 \ominus k_2))
\end{aligned}
$$

Indeed, given these, we have that

$$
\begin{aligned}
(k_1 \ominus k_2) \ominus (1 \ominus k_3) &= (k_1 \ominus k_2) \ominus ((k_1 \ominus k_2) \ominus k_3) && \text{(by } \mathsf{id}_{22}\text{)}\\
&= (k_1 \ominus k_2) \ominus ((k_1 \ominus k_2) \ominus (k_3 \ominus k_2)) && \text{(by } \mathsf{id}_{23}\text{)}\\
&= (k_1 \ominus k_2) \ominus (1 \ominus (k_3 \ominus k_2)) && \text{(by } \mathsf{id}_{22}\text{)}\\
&= k_1 \ominus (1 \ominus (k_3 \ominus k_2)). && \text{(by } \mathsf{id}_{24}\text{)}
\end{aligned}
$$

It remains to verify that id_{22}, $\mathsf{id}_{23,}$ and id_{24} are implied by $\mathcal{A}'$. We start by verifying $\mathcal{A}' \models \mathsf{id}_{22}$:

$$
\begin{aligned}
k_1 \ominus (1 \ominus k_2) &= k_1 \otimes (1 \ominus (1 \ominus k_2)) && \text{(by } \mathsf{id}_{11}\text{)}\\
&= (1 \ominus (1 \ominus k_2)) \otimes k_1 && \text{(by } \mathsf{id}_5\text{)}\\
&= (1 \ominus (1 \ominus k_2)) \otimes ((k_1 \ominus (1 \ominus (1 \ominus k_2)))\\
&\qquad \oplus (k_1 \ominus (k_1 \ominus (1 \ominus (1 \ominus k_2))))) && \text{(by } \mathsf{id}_4 \text{ and } \mathsf{id}_{12}\text{)}\\
&= ((1 \ominus (1 \ominus k_2)) \otimes (k_1 \ominus (1 \ominus (1 \ominus k_2))))\\
&\qquad \oplus ((1 \ominus (1 \ominus k_2)) \otimes (k_1 \ominus (k_1 \ominus (1 \ominus (1 \ominus k_2))))) && \text{(by } \mathsf{id}_8\text{)}\\
&= (k_1 \otimes ((1 \ominus (1 \ominus k_2)) \ominus (1 \ominus (1 \ominus k_2))))\\
&\qquad \oplus ((1 \ominus (1 \ominus k_2)) \otimes (k_1 \ominus (k_1 \ominus (1 \ominus (1 \ominus k_2))))) && \text{(by } \mathsf{id}_4 \text{ and } \mathsf{id}_{11}\text{)}\\
&= 0 \oplus ((1 \ominus (1 \ominus k_2)) \otimes (k_1 \ominus (k_1 \ominus (1 \ominus (1 \ominus k_2))))) && \text{(by } \mathsf{id}_2 \text{ and } \mathsf{id}_9\text{)}\\
&= (k_1 \otimes (1 \ominus (1 \ominus k_2))) \ominus (k_1 \ominus (1 \ominus (1 \ominus k_2))) && \text{(by } \mathsf{id}_3\text{, } \mathsf{id}_5\text{, and } \mathsf{id}_{11}\text{)}\\
&= (k_1 \otimes (1 \ominus (1 \ominus k_2))) \ominus (k_1 \otimes (1 \ominus (1 \ominus (1 \ominus k_2)))) && \text{(by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}\text{)}\\
&= (k_1 \otimes (1 \ominus (1 \ominus k_2))) \ominus (k_1 \ominus k_2) && \text{(by } \mathsf{id}_1\text{, } \mathsf{id}_{11,} \text{ and } \mathsf{id}_{16}\text{)}\\
&= 0 \oplus ((k_1 \otimes (1 \ominus (1 \ominus k_2))) \ominus (k_1 \ominus k_2)) && \text{(by } \mathsf{id}_3\text{)}\\
&= ((k_1 \ominus k_2) \ominus (k_1 \ominus k_2)) \oplus ((k_1 \otimes (1 \ominus (1 \ominus k_2))) \ominus (k_1 \ominus k_2)) && \text{(by } \mathsf{id}_9\text{)}\\
&= ((k_1 \ominus k_2) \oplus (k_1 \ominus (1 \ominus k_2))) \ominus (k_1 \ominus k_2) && \text{(by } \mathsf{id}_{11} \text{ and } \mathsf{id}_{14}\text{)}\\
&= (k_1 \otimes ((1 \ominus k_2) \oplus (1 \ominus (1 \ominus k_2)))) \ominus (k_1 \ominus k_2) && \text{(by } \mathsf{id}_8 \text{ and } \mathsf{id}_{11}\text{)}\\
&= (k_1 \otimes 1) \ominus (k_1 \ominus k_2) && \text{(by } \mathsf{id}_4 \text{ and } \mathsf{id}_{12}\text{)}\\
&= k_1 \ominus (k_1 \ominus k_2). && \text{(by } \mathsf{id}_{11}\text{)}
\end{aligned}
$$

To show $\mathcal{A}' \models \mathsf{id}_{23}$, we proceed as follows. Assume that

$$\begin{array}{ll} \mathsf{id}_{25}: & k_1 \ominus (k_2 \ominus k_1) = k_1 \\ \mathsf{id}_{26}: & k_1 \ominus (k_2 \ominus (k_3 \ominus k_1)) = k_1 \ominus k_2 \end{array}$$

is implied by $\mathcal{A}'$. Then,

$$\begin{aligned} (k_1 \ominus k_2) \ominus k_3 &= (k_1 \ominus k_2) \ominus (k_3 \ominus (k_2 \ominus (k_1 \ominus k_2))) && \text{(by } \mathsf{id}_{26}\text{)} \\ &= (k_1 \ominus k_2) \ominus ((k_3 \ominus (k_2 \ominus (k_1 \ominus k_2))) \ominus 0) && \text{(by } \mathsf{id}_{13}\text{)} \\ &= (k_1 \ominus k_2) \ominus ((k_3 \ominus (k_2 \ominus (k_1 \ominus k_2))) \ominus (k_2 \ominus (k_2 \ominus (k_1 \ominus k_2)))) && \text{(by } \mathsf{id}_{20}\text{)} \\ &= (k_1 \ominus k_2) \ominus (k_3 \ominus ((k_2 \ominus (k_1 \ominus k_2)) \oplus (k_2 \ominus (k_2 \ominus (k_1 \ominus k_2))))) && \text{(by } \mathsf{id}_{10}\text{)} \\ &= (k_1 \ominus k_2) \ominus (k_3 \ominus k_2). && \text{(by } \mathsf{id}_4 \text{ and } \mathsf{id}_{12}\text{)} \end{aligned}$$

The identity id_{25} is indeed implied by $\mathcal{A}'$:

$$\begin{aligned} k_1 &= (k_1 \ominus (k_2 \ominus k_1)) \oplus (k_1 \ominus (k_1 \ominus (k_2 \ominus k_1))) && \text{(by } \mathsf{id}_4 \text{ and } \mathsf{id}_{12}\text{)} \\ &= (k_1 \ominus (k_2 \ominus k_1)). && \text{(by } \mathsf{id}_3 \text{ and } \mathsf{id}_{20}\text{)} \end{aligned}$$

The same holds for the identity id_{26}:

$$\begin{aligned} k_1 \ominus k_2 &= k_1 \ominus ((k_2 \ominus (1 \ominus (k_3 \ominus k_1))) \oplus (k_2 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1))))) && \text{(by } \mathsf{id}_4 \text{ and } \mathsf{id}_{12}\text{)} \\ &= (k_1 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) \ominus (k_2 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) && \text{(by } \mathsf{id}_{10}\text{)} \\ &= (((k_1 \ominus (k_3 \ominus k_1)) \oplus (k_1 \ominus (k_1 \ominus (k_3 \ominus k_1)))) \\ &\qquad \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) \ominus (k_2 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) && \text{(by } \mathsf{id}_4 \text{ and } \mathsf{id}_{12}\text{)} \\ &= (((k_1 \ominus (k_3 \ominus k_1)) \oplus 0) \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) \\ &\qquad \ominus (k_2 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) && \text{(by } \mathsf{id}_{20}\text{)} \\ &= ((k_1 \ominus (k_3 \ominus k_1)) \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) \\ &\qquad \ominus (k_2 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) && \text{(by } \mathsf{id}_3\text{)} \\ &= (k_1 \otimes ((1 \ominus (k_3 \ominus k_1)) \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1))))) \\ &\qquad \ominus (k_2 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) && \text{(by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}\text{)} \\ &= (k_1 \otimes (1 \ominus (k_3 \ominus k_1))) \ominus (k_2 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) && \text{(by } \mathsf{id}_{25}\text{)} \\ &= (k_1 \ominus (k_3 \ominus k_1)) \ominus (k_2 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) && \text{(by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}\text{)} \\ &= k_1 \ominus (k_2 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) && \text{(by } \mathsf{id}_{25}\text{)} \\ &= k_1 \ominus (k_2 \ominus (k_2 \ominus (k_2 \ominus (k_3 \ominus k_1)))) && \text{(by } \mathsf{id}_{22}\text{)} \\ &= k_1 \ominus (k_2 \ominus (k_3 \ominus k_1)). && \text{(by } \mathsf{id}_{16}\text{)} \end{aligned}$$

We next show that $\mathcal{A}' \models \mathsf{id}_{24}$, which states that $(k_1 \ominus k_2) \ominus (1 \ominus (k_3 \ominus k_2)) = k_1 \ominus (1 \ominus (k_3 \ominus k_2))$ for all k_1, k_2, and k_3 in K. Assume that

$$\mathsf{id}_{27}: \quad k_1 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1))) = k_1$$

is implied by $\mathcal{A}'$. Then, $\mathcal{A}' \models \mathsf{id}_{24}$:

$$\begin{aligned} (k_1 \ominus k_2) \ominus (1 \ominus (k_3 \ominus k_2)) &= (k_1 \ominus (k_2 \ominus (1 \ominus k_2))) \ominus (1 \ominus (k_3 \ominus k_2)) && \text{(by } \mathsf{id}_{25}\text{)} \\ &= (k_1 \ominus (k_2 \ominus (1 \ominus (k_2 \ominus (k_1 \ominus (1 \ominus (k_3 \ominus k_2)))))))) \ominus (1 \ominus (k_3 \ominus k_2)) \\ &\qquad && \text{(by } \mathsf{id}_{27}\text{)} \\ &= (k_1 \ominus (1 \ominus (k_3 \ominus k_2))) \ominus (k_2 \ominus (1 \ominus (k_2 \ominus (k_1 \ominus (1 \ominus (k_3 \ominus k_2)))))) \\ &\qquad && \text{(by } \mathsf{id}_4 \text{ and } \mathsf{id}_{10}\text{)} \\ &= k_1 \ominus (1 \ominus (k_3 \ominus k_2)). && \text{(by } \mathsf{id}_{27}\text{)} \end{aligned}$$

The identity id_{27} is indeed implied by $\mathcal{A}'$:

$$\begin{aligned}
k_1 &= k_1 \ominus (k_3 \ominus k_1) && (\text{by } \mathsf{id}_{25})\\
&= k_1 \otimes (1 \ominus (k_3 \ominus k_1)) && (\text{by } \mathsf{id}_{11})\\
&= k_1 \otimes ((1 \ominus (k_3 \ominus k_1)) \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1)))) && (\text{by } \mathsf{id}_{25})\\
&= (k_1 \ominus (k_3 \ominus k_1)) \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1))) && (\text{by } \mathsf{id}_{11})\\
&= k_1 \ominus (k_2 \ominus (1 \ominus (k_3 \ominus k_1))) && (\text{by } \mathsf{id}_{25})
\end{aligned}$$

Finally, we verify that $\mathcal{A}'$ implies $\mathsf{id}_{18,}$ which states that $k_1 \ominus (k_2 \oplus k_2) = k_1 \ominus k_2$ for all k_1, k_2 in K.

$$\begin{aligned}
k_1 \ominus k_2 &= (k_1 \ominus k_2) \otimes 1 && (\text{by } \mathsf{id}_1)\\
&= (k_1 \ominus k_2) \otimes ((1 \ominus k_2) \oplus (1 \ominus (1 \ominus k_2))) && (\text{by } \mathsf{id}_4 \text{ and } \mathsf{id}_{12})\\
&= ((k_1 \ominus k_2) \ominus k_2) \oplus ((k_1 \ominus k_2) \ominus (1 \ominus k_2)) && (\text{by } \mathsf{id}_1, \mathsf{id}_8, \text{ and } \mathsf{id}_{11})\\
&= ((k_1 \ominus k_2) \ominus k_2) \oplus (k_1 \ominus (k_2 \oplus (1 \ominus k_2))) && (\text{by } \mathsf{id}_{10})\\
&= ((k_1 \ominus k_2) \ominus k_2) \oplus (k_1 \otimes (1 \ominus (k_2 \oplus (1 \ominus k_2)))) && (\text{by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11})\\
&= ((k_1 \ominus k_2) \ominus k_2) \oplus (k_1 \otimes ((1 \ominus k_2) \ominus (1 \ominus k_2))) && (\text{by } \mathsf{id}_{10})\\
&= ((k_1 \ominus k_2) \ominus k_2) \oplus (k_1 \otimes 0) && (\text{by } \mathsf{id}_9)\\
&= k_1 \ominus (k_2 \oplus k_2). && (\text{by } \mathsf{id}_2, \mathsf{id}_3, \text{ and } \mathsf{id}_{10})
\end{aligned}$$

Hence, $\mathcal{A}' \models \{\mathsf{id}_1, \ldots, \mathsf{id}_{18}\}$. We are now ready to prove the three claims.

Proof of Claim 1. We first check whether the definitions of disjunction, conjunction, and complementation are independent of the choice of elements in K and thus that these operations are well-defined. Since conjunction is defined in terms of disjunction and complementation, it suffices to consider disjunction and complementation only. Let k, k_1, k_2, ℓ, ℓ_1, and ℓ_2 be elements in K such that $b = d(k) = d(\ell)$, $b_1 = d(k_1) = d(\ell_1)$, and $b_2 = d(k_2) = d(\ell_2)$. We start by showing that complementation is well-defined:

$$\begin{aligned}
\bar{b} &:= 1 \ominus k && (\text{by def.})\\
&= 1 \ominus (1 \ominus (1 \ominus k)) && (\text{by } \mathsf{id}_{16})\\
&= 1 \ominus (1 \ominus (1 \ominus \ell)) && (\text{by } d(k) = d(\ell))\\
&= 1 \ominus \ell. && (\text{by } \mathsf{id}_{16})
\end{aligned}$$

For disjunction, we observe the following:

$$\begin{aligned}
b_1 \vee b_2 &:= 1 \ominus (1 \ominus (k_1 \oplus k_2)) && (\text{by def.})\\
&= 1 \ominus ((1 \ominus k_1) \ominus k_2) && (\text{by } \mathsf{id}_{10})\\
&= 1 \ominus ((1 \ominus \ell_1) \ominus k_2) && (\text{by } d(k) = d(\ell))\\
&= 1 \ominus (1 \ominus (\ell_1 \oplus k_2)) && (\text{by } \mathsf{id}_{10})\\
&= 1 \ominus (1 \ominus (k_2 \oplus \ell_1)) && (\text{by } \mathsf{id}_4)\\
&= 1 \ominus ((1 \ominus k_2) \ominus \ell_1) && (\text{by } \mathsf{id}_{10})\\
&= 1 \ominus ((1 \ominus \ell_2) \ominus \ell_1) && (\text{by } d(k) = d(\ell))\\
&= 1 \ominus (1 \ominus (\ell_2 \oplus \ell_1)) && (\text{by } \mathsf{id}_{10})\\
&= 1 \ominus (1 \ominus (\ell_1 \oplus \ell_2)). && (\text{by } \mathsf{id}_4)
\end{aligned}$$

Hence, $\bar{}$ and $\vee$ are well-defined and so is $\wedge$.

$$\begin{array}{ll} \mathsf{b}_1\colon & b_1 \vee b_2 = b_2 \vee b_1 \\ \mathsf{b}_2\colon & b_1 \vee (b_2 \vee b_3) = (b_1 \vee b_2) \vee b_3 \\ \mathsf{b}_3\colon & b_1 = (b_1 \wedge b_2) \vee (b_1 \wedge \overline{b_2}) \end{array}$$

Fig. 6. Axiomatization of Boolean algebras.

We next verify that $(B, \vee, \wedge, \bar{}, \bot, \top)$ is indeed a Boolean algebra. We do this by showing that this structure satisfies the three identities b_1, b_2, and b_3 (the "fourth set" of postulates, [Huntington 1933b, page 280] and corrected in Huntington [1933a]) shown in Figure 6. The commutativity (b_1) of the disjunction ($\vee$) follows immediately from the definition of disjunction and the fact that $\oplus$ is commutative (id_4). For associativity (b_2), we observe the following. Let b_1, b_2, and b_3 in B and k_1, k_2, and k_3 in K be such that $b_i = d(k_i)$, for $i = 1, 2, 3$.

$$\begin{aligned} b_1 \vee (b_2 \vee b_3) &:= 1 \ominus (1 \ominus (k_1 \oplus (1 \ominus (1 \ominus (k_2 \oplus k_3))))) && \text{(by def.)} \\ &= 1 \ominus ((1 \ominus k_1) \ominus (1 \ominus (1 \ominus (k_2 \oplus k_3)))) && (\text{by } \mathsf{id}_{10}) \\ &= 1 \ominus ((1 \ominus k_1) \otimes (1 \ominus (1 \ominus (1 \ominus (k_2 \oplus k_3))))) && (\text{by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}) \\ &= 1 \ominus ((1 \ominus k_1) \otimes (1 \ominus (k_2 \oplus k_3))) && (\text{by } \mathsf{id}_{16}) \\ &= 1 \ominus ((1 \ominus k_1) \ominus (k_2 \oplus k_3)) && (\text{by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}) \\ &= 1 \ominus (1 \ominus (k_1 \oplus (k_2 \oplus k_3))) && (\text{by } \mathsf{id}_{10}) \\ &= 1 \ominus (1 \ominus ((k_1 \oplus k_2) \oplus k_3)) && (\text{by } \mathsf{id}_6) \\ &= 1 \ominus ((1 \ominus k_3) \ominus (k_1 \oplus k_2)) && (\text{by } \mathsf{id}_4 \text{ and } \mathsf{id}_{10}) \\ &= 1 \ominus ((1 \ominus k_3) \otimes (1 \ominus (k_1 \oplus k_2))) && (\text{by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}) \\ &= 1 \ominus ((1 \ominus k_3) \otimes (1 \ominus (1 \ominus (1 \ominus (k_1 \oplus k_2))))) && (\text{by } \mathsf{id}_{16}) \\ &= 1 \ominus ((1 \ominus k_3) \ominus (1 \ominus (1 \ominus (k_1 \oplus k_2)))) && (\text{by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}) \\ &= 1 \ominus (1 \ominus ((1 \ominus (1 \ominus (k_1 \oplus k_2))) \oplus k_3)) && (\text{by } \mathsf{id}_4 \text{ and } \mathsf{id}_{10}) \\ &:= (b_1 \vee b_2) \vee b_3 && \text{(by def.)} \end{aligned}$$

It remains to verify (b_3). Let $k_1, k_2 \in K$ and $b_1, b_2 \in B$ such that $b_1 = d(k_1)$ and $b_2 = d(k_2)$. Recall that, by definition, $b_1 \wedge b_2 := \overline{(\bar{b}_1 \vee \bar{b}_2)} \vee \overline{(\bar{b}_1 \vee b_2)}$. Let us first consider $\overline{(\bar{b}_1 \vee \bar{b}_2)}$ and $\overline{\bar{b}_1 \vee b_2}$.

$$\begin{aligned} \overline{(\bar{b}_1 \vee \bar{b}_2)} &= 1 \ominus (1 \ominus (1 \ominus ((1 \ominus (1 \ominus (1 \ominus k_1))) \oplus (1 \ominus (1 \ominus (1 \ominus k_2)))))) && \text{(by def.)} \\ &= 1 \ominus (1 \ominus (1 \ominus ((1 \ominus k_1) \oplus (1 \ominus k_2)))) && (\text{by } \mathsf{id}_{16}) \\ &= 1 \ominus ((1 \ominus k_1) \oplus (1 \ominus k_2)), && (\text{by } \mathsf{id}_{16}) \end{aligned}$$

and similarly,

$$\overline{\bar{b}_1 \vee b_2} = 1 \ominus ((1 \ominus k_1) \oplus (1 \ominus (1 \ominus k_2))).$$

Hence, $\overline{(\bar{b}_1 \vee \bar{b}_2)} \vee \overline{(\bar{b}_1 \vee b_2)}$ is equal to

$$\begin{aligned} & 1 \ominus (1 \ominus ((1 \ominus ((1 \ominus k_1) \oplus (1 \ominus k_2))) \oplus (1 \ominus ((1 \ominus k_1) \oplus (1 \ominus (1 \ominus k_2)))))) && \text{(by def.)} \\ &= 1 \ominus (1 \ominus (((1 \ominus (1 \ominus k_2)) \ominus (1 \ominus k_1)) \oplus ((1 \ominus (1 \ominus (1 \ominus k_2))) \ominus (1 \ominus k_1)))) \\ & && (\text{by } \mathsf{id}_4 \text{ and } \mathsf{id}_{10}) \\ &= 1 \ominus (1 \ominus (((1 \ominus (1 \ominus k_2)) \ominus (1 \ominus k_1)) \oplus ((1 \ominus k_2) \ominus (1 \ominus k_1)))) && (\text{by } \mathsf{id}_{16}) \\ &= 1 \ominus (1 \ominus ((((1 \ominus (1 \ominus k_2)) \oplus (1 \ominus k_2)) \ominus (1 \ominus k_1)))) && (\text{by } \mathsf{id}_{14}) \end{aligned}$$

$$= 1 \ominus (1 \ominus (1 \ominus (1 \ominus k_1))) \qquad \text{(by } \mathsf{id}_{12}\text{)}$$
$$= 1 \ominus (1 \ominus k_1), \qquad \text{(by } \mathsf{id}_{16}\text{)}$$

which in turn is equal to b_1.

Hence, $\vee$ and $\bar{}$ satisfy b_1–b_3. It has been shown in Huntington [1933b] that when $\wedge$ is defined in terms of $\vee$ and $\bar{}$, as we do, then $\top$ is uniquely determined by $\top = b \vee \bar{b}$. In addition, $\bot := \bar{\top}$. It is known that these operations and constants define a Boolean algebra [Huntington 1933b]. Since we defined $\top := d(1)$ and $\bot := d(0)$, it remains to be shown that $d(1) = b \vee \bar{b}$ for any $b \in B$ and that $d(0) = \overline{d(1)}$. Observe the following: Let $b \in B$ and $k \in K$ such that $b = d(k)$. Then,

$$\begin{aligned} b \vee \bar{b} &:= 1 \ominus (1 \ominus (k \oplus (1 \ominus k))) && \text{(by def.)} \\ &= 1 \ominus ((1 \ominus k) \ominus (1 \ominus k)) && \text{(by } \mathsf{id}_{10}\text{)} \\ &= 1 \ominus 0 && \text{(by } \mathsf{id}_{9}\text{)} \\ &= 1 \ominus (1 \ominus 1) && \text{(by } \mathsf{id}_{9}\text{)} \\ &= d(1) && \text{(by def.)} \end{aligned}$$

Furthermore, $\overline{d(1)} = 1 \ominus (1 \ominus (1 \ominus 1)) = 1 \ominus (1 \ominus 0) = d(0)$ by id_9 and the definition of d. Hence, we may conclude that $(B, \vee, \wedge, \bar{}, \bot, \top)$ is indeed a boolean algebra. This concludes the proof of Claim 1. □

Proof of Claim 2. We need to verify that (K, B, d, ι) satisfies sb_1–sb_7.

Since we defined $\bot := d(0)$ and $\top := d(1)$, sb_1 is automatically satisfied. For sb_2, we observe that $\iota(\bot) = \iota(d(0)) = d(0) = 0$ and $\iota(\top) = \iota(d(1)) = d(1) = 1$. This follows from the definition of d and ι, and identities id_9 and id_{13}.

We next show that sb_3 holds; that is, for any $k, \ell \in K$, $d(k \oplus \ell) = d(k) \vee d(\ell)$. Observe that $d(k \oplus \ell) = 1 \ominus (1 \ominus (k \oplus \ell))$ which is equal to $d(k) \vee d(\ell)$ by the definition of d and $\vee$.

We verify sb_5; that is, for any $k \in K$, $b \in B$, $d(k \otimes \iota(b)) = d(k) \wedge b$, as follows. Let $\ell \in K$ be such that $b = d(\ell)$. Then,

$$\begin{aligned} d(k \otimes (1 \ominus (1 \ominus \ell))) &= 1 \ominus (1 \ominus (k \otimes (1 \ominus (1 \ominus \ell)))) && \text{(by def.)} \\ &= 1 \ominus (1 \ominus (k \ominus (1 \ominus \ell))) && \text{(by } \mathsf{id}_1 \text{ and } \mathsf{id}_{11}\text{)} \\ &= (1 \ominus (1 \ominus \ell)) \ominus (1 \ominus k) && \text{(by } \mathsf{id}_{17}\text{)} \\ &= 1 \ominus ((1 \ominus k) \oplus (1 \ominus \ell)) && \text{(by } \mathsf{id}_4 \text{ and } \mathsf{id}_{10}\text{)} \\ &= \overline{\overline{d(k)} \vee \overline{d(\ell)}} && \text{(by } \mathsf{id}_{16} \text{ and def.)} \\ &= d(k) \wedge b. && \text{(by def.)} \end{aligned}$$

For sb_4, we need to show that for any $b, b' \in B$, $\iota(b \vee b') = \iota(b) \oplus (\iota(\bar{b}) \otimes \iota(b'))$. Since, ι is the identity mapping, this is equivalent to showing that for any $k, \ell \in K$, $d(k) \vee d(\ell) = d(k) \oplus (\overline{d(k)} \otimes d(\ell))$.

$$\begin{aligned} d(k) \vee d(\ell) &= 1 \ominus (1 \ominus (k \oplus \ell)) && \text{(by def.)} \\ &= 1 \ominus ((1 \ominus k) \otimes (1 \ominus \ell)) && \text{(by } \mathsf{id}_{15}\text{)} \\ &= ((1 \ominus k) \oplus (1 \ominus (1 \ominus k))) \ominus ((1 \ominus k) \otimes (1 \ominus \ell)) && \text{(by } \mathsf{id}_{12}\text{)} \\ &= ((1 \ominus k) \ominus ((1 \ominus k) \otimes (1 \ominus \ell))) \oplus ((1 \ominus (1 \ominus k)) \ominus ((1 \ominus k) \otimes (1 \ominus \ell))) \\ & && \text{(by } \mathsf{id}_{14}\text{)} \\ &= ((1 \ominus k) \ominus (1 \ominus \ell)) \oplus ((1 \ominus (1 \ominus k)) \ominus ((1 \ominus k) \otimes (1 \ominus \ell))) && \text{(by } \mathsf{id}_{16}\text{)} \\ &= ((1 \ominus (k \oplus (1 \ominus \ell)))) \oplus ((1 \ominus (1 \ominus k)) \ominus ((1 \ominus k) \otimes (1 \ominus \ell))) \\ & && \text{(by } \mathsf{id}_1, \mathsf{id}_{11}, \text{ and } \mathsf{id}_{12}\text{)} \end{aligned}$$

$$
\begin{aligned}
&= ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))) \oplus ((1 \ominus (1 \ominus k)) \ominus ((1 \ominus k) \otimes (1 \ominus \ell))) && \text{(by } \mathsf{id}_{15}\text{)}\\
&= ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))) \oplus (1 \ominus ((1 \ominus k) \oplus ((1 \ominus k) \otimes (1 \ominus \ell)))) && \text{(by } \mathsf{id}_{10}\text{)}\\
&= ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))) \oplus (1 \ominus ((1 \ominus k) \otimes (1 \oplus (1 \ominus \ell)))) && \text{(by } \mathsf{id}_{8}\text{)}\\
&= ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))) \oplus (1 \ominus ((1 \ominus k) \otimes (((1 \ominus \ell)\\
&\qquad \oplus (1 \ominus (1 \ominus \ell))) \oplus (1 \ominus \ell)))) && \text{(by } \mathsf{id}_{4} \text{ and } \mathsf{id}_{12}\text{)}\\
&= ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))) \oplus (1 \ominus (((1 \ominus k) \otimes (1 \ominus \ell))\\
&\qquad \oplus ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))) \oplus ((1 \ominus k) \otimes (1 \ominus \ell)))) && \text{(by } \mathsf{id}_{8}\text{)}\\
&= ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))) \oplus (1 \ominus (((1 \ominus k) \otimes (1 \ominus \ell))\\
&\qquad \oplus ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))))) && \text{(by } \mathsf{id}_{4}, \mathsf{id}_{10,} \text{ and } \mathsf{id}_{18}\text{)}\\
&= ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))) \oplus (1 \ominus ((1 \ominus k) \otimes ((1 \ominus \ell) \oplus (1 \ominus (1 \ominus \ell))))) && \text{(by } \mathsf{id}_{8}\text{)}\\
&= ((1 \ominus k) \otimes (1 \ominus (1 \ominus \ell))) \oplus (1 \ominus (1 \ominus k)) && \text{(by } \mathsf{id}_{1}, \mathsf{id}_{4}, \text{ and } \mathsf{id}_{12}\text{)}\\
&= d(k) \oplus (\overline{d(k)} \otimes d(\ell)) && \text{(by } \mathsf{id}_{4}, \mathsf{id}_{16,} \text{ and def.)}
\end{aligned}
$$

For sb_6, we need to show that for any $b, b' \in B$, $\iota(b \wedge b') = \iota(b) \otimes \iota(b')$. Since ι is the identity mapping, this is equivalent to showing that, for any $k, \ell \in K$, $d(k) \wedge d(\ell) = d(k) \otimes d(\ell)$:

$$
\begin{aligned}
d(k) \wedge d(\ell) &= 1 \ominus ((1 \ominus k) \oplus (1 \ominus \ell)) && \text{(by def.)}\\
&= (1 \ominus (1 \ominus k)) \otimes (1 \ominus (1 \ominus \ell)) && \text{(by } \mathsf{id}_{1}, \mathsf{id}_{10,} \text{ and } \mathsf{id}_{11}\text{)}\\
&= d(k) \otimes d(\ell) && \text{(by def.)}
\end{aligned}
$$

Finally, for sb_7 we verify that for any $k \in K$, $k \otimes \overline{d(k)} = 0$. It is indeed readily verified that

$$k \otimes \overline{d(k)} = k \otimes (1 \ominus (1 \ominus (1 \ominus k))) = k \otimes (1 \ominus k) = k \ominus k = 0. \qquad \text{(by } \mathsf{id}_{1}, \mathsf{id}_{11,} \text{ and } \mathsf{id}_{16}\text{)}$$

In other words, (K, B, d, ι) is indeed a seba-structure. This concludes the proof of Claim 2. □

Proof of Claim 3. We need to verify that $(K, \oplus, \otimes, \ominus, 0, 1)$ is derived from (K, B, d, ι). For this, we need to verify that $k_1 \ominus k_2 = k_1 \otimes \iota(\overline{d(k_2)})$ for all $k_1, k_2 \in K$. Observe the following:

$$
\begin{aligned}
k_1 \ominus k_2 &= k_1 \otimes (1 \ominus k_2) && \text{(by } \mathsf{id}_{1} \text{ and } \mathsf{id}_{11}\text{)}\\
&= k_1 \otimes (1 \ominus (1 \ominus (1 \ominus k_2))) && \text{(by } \mathsf{id}_{16}\text{)}\\
&= k_1 \otimes \overline{d(k_2)}. && \text{(by def.)}
\end{aligned}
$$

This concludes the proof of Claim 3. □
This concludes the proof of Theorem 5.3. □

Proof of Lemma 6.8

Let S be a set of monomials and consider $(P, N) \in \mathcal{C}(S)$ and a pair (P', N') whose monomials belong to S. We first show that

$$[\mathbf{b}_{P,N}] \cdot [\mathbf{b}_{P',N'}] = \begin{cases} [\mathbf{b}_{P,N}] & \text{if } s(P') \subseteq P \text{ and } s(N') \subseteq N \\ [0] & \text{otherwise.} \end{cases}$$

Here, $s(P')$ and $s(N')$ denote the "set versions" of P' and N', respectively. Recall that $[\mathbf{b}_{P',N'}] = [\mathbf{b}_{s(P'),s(N')}]$ because of the idempotence of "$\cdot$" for variables in B_X and $\bar{B}_X$. We can thus safely use $\mathbf{b}_{s(P'),s(N')}$ instead of $\mathbf{b}_{P',N'}$. Since $(P, N) \in \mathcal{C}(S)$, every monomial μ in S either appears in P or in N.

First, assume that $s(P') \subseteq P$ and $s(N') \subseteq N$. Then, every monomial in $s(P')$ already appears in P and similarly for $s(N')$ and N. The idempotence of "$\cdot$" implies that $[\mathbf{b}_{P,N}] \cdot [\mathbf{b}_{s(P'),s(N')}] = [\mathbf{b}_{P,N}]$. Suppose next that $s(P')$ contains a monomial μ that does not appear in P. This implies that μ appears in N and thus $[\mathbf{b}_{P,N}] \cdot [\mathbf{b}_{s(P'),s(N')}]$ contains $\bar{b}_\mu \cdot b_\mu$ which, by the properties of θ, is equivalent to 0. Hence, also $[\mathbf{b}_{P,N}] \cdot [\mathbf{b}_{s(P'),s(N')}] = [0]$. The case that $s(N') \not\subseteq N$ can be dealt with similarly.

We next show that

$$\left[\sum_{(P,N)\in\mathcal{C}(S)} \mathbf{b}_{P,N}\right] = [1]$$

by induction on the size of S. For the base case, assume that S consists of a single monomial μ. Then, $\mathcal{C}(S) = \{(\{\mu\}, \emptyset), (\emptyset, \{\mu\})\}$, and thus the sum corresponds to $[b_\mu + \bar{b}_\mu]$ which, by the properties of θ is equal to [1]. Suppose next that the property holds for sets S of size k. Consider $S' = S \cup \{\mu\}$ for some monomial $\mu \notin S$. Note that any $(P', N') \in \mathcal{C}(S')$ can be written either as $(P \cup \{\mu\}, N')$ with $(P, N') \in \mathcal{C}(S)$ or as $(P', N \cup \{\mu\})$ with $(P', N) \in \mathcal{C}(S)$. Similarly, for $(P, N) \in \mathcal{C}(S)$, we have that either $(P \cup \{\mu\}, N) \in \mathcal{C}(S')$, provided that $[\mathbf{b}_{P\cup\{\mu\},N}] \neq [0]$, or $(P, N \cup \{\mu\}) \in \mathcal{C}(S')$, provided that $[\mathbf{b}_{P,N\cup\{\mu\}}] \neq [0]$. We can thus write

$$\begin{aligned}\left[\sum_{(P',N')\in\mathcal{C}(S')} \mathbf{b}_{P,N}\right] &= \left[\sum_{(P,N)\in\mathcal{C}(S)} b_\mu \cdot \mathbf{b}_{P,N}\right] + \left[\sum_{(P,N)\in\mathcal{C}(S)} \bar{b}_\mu \cdot \mathbf{b}_{P,N}\right] \\ &= \left[(b_\mu + \bar{b}_\mu) \cdot \sum_{(P,N)\in\mathcal{C}(S)} \mathbf{b}_{P,N}\right] = \left[\sum_{(P,N)\in\mathcal{C}(S)} \mathbf{b}_{P,N}\right] = [1],\end{aligned}$$

where the last equality follows from the induction hypothesis and the fact that $b_\mu + \bar{b}_\mu$ is equivalent to 1. □

Proof of Proposition 6.14

We show that every polynomial $p[X_B]$ in $\mathbb{N}[X_B]$ is equivalent to its expanded version $p_e[X_B]$. Assume that $p[X_B] = \sum_{(P,N)\in T} \mathbf{b}_{P,N} \cdot p_{P,N}[X]$ for some $T \subseteq \mathcal{P}N$ and non-zero polynomials $p_{P,N}[X] \in \mathbb{N}[X]$. It is easily verified that any polynomial in $\mathbb{N}[X_B]$ can be written in this form. Recall that P and N are, in general, multisets of monomials. Denote by $s(P)$ and $s(N)$ their corresponding set versions obtained by ignoring multiplicities of the monomials in P and N, respectively.

Consider the set of monomials S_p obtained from the polynomial $p[X_B]$ (cf. Definition 6.9). Observe that S_p contains all monomials μ that appear as b_μ or $\bar{b}_\mu$ in pairs $(P, N) \in T$ and those that appear in any of the polynomials $p_{P,N}[X]$. In other words, S_p and T are defined over the same set of monomials. We construct $p_e[X_B]$ in two steps, as illustrated in Example 6.11.

Step 1. We first multiply $p[X_B]$ with $\sum_{(P,N)\in\mathcal{C}(S_p)} \mathbf{b}_{P,N}$, which by Property (2) in Lemma 6.8 is equivalent to [1], and we simplify the terms, using Property (1) in Lemma 6.8, just as in Example 6.11. The general form of the resulting polynomial $p_1[X_B]$ is then given by

$$p_1[X_B] = \sum_{(P',N')\in\mathcal{C}(S_p)\setminus U_1} \mathbf{b}_{P',N'} \cdot \left(\sum_{(P,N)\in T, s(P)\subseteq P', s(N)\subseteq N'} p_{P,N}[X]\right),$$

where U_1 denotes the set of pairs $(P', N') \in \mathcal{C}(S_p)$ for which $\{(P, N) \in T, s(P) \subseteq P', s(N) \subseteq N'\}$ is empty. In terms of equivalence classes, it is safe to eliminate the terms corresponding to U_1 because these are equivalent to 0 by Property (1) in Lemma 6.8.

Step 2. Next, for $(P', N') \in \mathcal{C}(S_p) \setminus U_1$, denote by $p'_{P,N}[X]$ the polynomial in $\mathbb{N}[X]$ obtained by removing all monomials μ from $p_{P,N}[X]$ such that $\mu \in N$. If no monomials are left in $p'_{P,N}[X]$, we treat it as the zero polynomial. Furthermore, let $r_{P',N'}[X] = \sum_{(P,N)\in T, s(P)\subseteq P', s(N)\subseteq N'} p'_{P,N}[X]$. We can then write $p_1[X_B]$ equivalently as

$$p_e[X_B] = \sum_{(P',N')\in\mathcal{C}(S_p)\setminus(U_1\cup U_2)} \mathbf{b}_{P',N'} \cdot r_{P',N'}[X],$$

where $U_2 \subseteq \mathcal{C}(S_p) \setminus U_1$ consists of all (P', N') for which $r_{P',N'}[X] = 0$. In terms of equivalence classes, it is safe to eliminate the terms corresponding to U_2 because this corresponds to the application of $[\bar{b}_\mu] \cdot [\mu] = [0]$.

We next verify that $p_e[X_B]$ is the expanded version of $p[X_B]$ as defined in Definition 6.15. For this, we need to show that (i) the polynomials $r_{P',N'}[X]$ are non-zero and do not contain monomials that occur in N', and (ii) $\mathcal{C}(S_p) \setminus (U_1 U_2) = S_p\text{-supp}(p[X_B])$.

Clearly, (i) follows from the construction of $p_e[X_B]$. We next verify (ii).

We first show that $S_p\text{-supp}(p[X_B]) \subseteq \mathcal{C}(S_p) \setminus (U_1 \cup U_2)$. Let $(P, N) \in \mathcal{C}(S_p)$ and assume that $(P, N) \in S_p\text{-supp}(p[X_B])$ but $(P, N) \in U_1 \cup U_2$. This is impossible, however. Indeed, $(P, N) \in S_p\text{-supp}(p[X_B])$ implies that $[0] \neq [\mathbf{b}_{P,N} \cdot p[X_B]] = [\mathbf{b}_{P,N} \cdot p_e[X_B]]$. On the other hand, $(P, N) \in (U_1 \cup U_2)$ implies that $[\mathbf{b}_{P,N} \cdot p_e[X_B]] = [0]$.

To show that $\mathcal{C}(S_p) \setminus (U_1 \cup U_2) \subseteq S_p\text{-supp}(p[X_B])$, let $(P', N') \in \mathcal{C}(S_p) \setminus (U_1 \cup U_2)$ and assume that $(P', N') \notin S_p\text{-supp}(p[X_B])$. The latter implies that $[\mathbf{b}_{P',N'} \cdot p[X_B]] = [0]$, and $(P', N') \in \mathcal{C}(S_p) \setminus (U_1 \cup U_2)$ implies that there exists a non-zero polynomial $r_{P',N'}[X]$ in $p_e[X_B]$. Hence, $[\mathbf{b}_{P',N'} \cdot r_{P',N'}[X]] = [0]$ and there must be a μ in $r_{P',N'}[X]$ that belongs to N'. This contradicts the assumption that $(P', N') \in \mathcal{C}(S_p) \setminus (U_1 \cup U_2)$. We may thus conclude that $\mathcal{C}(S_p) \setminus (U_1 \cup U_2) \subseteq S_p\text{-supp}(p[X_B])$. Hence, $\mathcal{C}(S_p) \setminus (U_1 \cup U_2) = S_p\text{-supp}(p[X_B])$.

We conclude by showing that $p_e[X_B]$ is uniquely determined by $p[X_B]$. Let $p'[X_B] = \sum_{(P',N')\in S_p\text{-supp}(p[X_B])} \mathbf{b}_{P',N'} \cdot s_{P',N'}[X]$ for some non-zero $s_{P',N'}[X] \in \mathbb{N}[X]$ none of whose monomials appears in N'. Assume that (a) $[p[X_B]] = [p'[X_B]]$ and (b) $p_e[X_B] \neq p'[X_B]$ where $p_e[X_B]$ is of the form (‡). From (a) we know that $[p_e[X_B]] = [p'[X_B]]$ and that $[\mathbf{b}_{P',N'} \cdot p_e[X_B]] = [\mathbf{b}_{P',N'} \cdot r_{P',N'}[X]] = [\mathbf{b}_{P',N'} \cdot s_{P',N'}[X]] = [\mathbf{b}_{P',N'} \cdot p'[X_B]]$. From (b), we know that there is a pair (P', N') in $S_p\text{-supp}(p[X_B])$ such that $r_{P',N'}[X] \neq s_{P',N'}[X]$. More specifically, let $r_{P',N'}[X] = \sum_\mu r_\mu \cdot \mu$ and $s_{P',N'}[X] = \sum_\mu s_\mu \cdot \mu$ and let μ be such that $r_\mu \neq s_\mu$. Without loss of generality, we may assume that $s_\mu > r_\mu$. To ensure that $[\mathbf{b}_{P',N'} \cdot r_{P',N'}[X]] = [\mathbf{b}_{P',N'} \cdot s_{P',N'}[X]]$, we thus must be able to go from $\mathbf{b}_{P',N'} \cdot r_{P',N'}[X]$ to $\mathbf{b}_{P',N'} \cdot s_{P',N'}[X]$ by applying a sequence of rules (i)–(iv) as given in the definition of θ. In particular, we must add $\mathbf{b}_{P',N'} \cdot (r_\mu - s_\mu) \cdot \mu$ to $\mathbf{b}_{P',N'} \cdot r_{P',N'}[X]$ to obtain $\mathbf{b}_{P',N'} \cdot s_{P',N'}[X]$. From the definition of θ, however, we can only introduce such a term using rule (iii). That is, whenever $\mu \in N$. This is impossible since $s_{P',N'}[X]$ is such that none of its monomials occurs in N'. Hence, $[\mathbf{b}_{P',N'} \cdot r_{P',N'}[X]] \neq [\mathbf{b}_{P',N'} \cdot s_{P',N'}[X]]$, which contradicts (i). Hence, $p_e[X_B]$ must be equal to $p'[X_B]$. □

Proof of Proposition 6.16

Suppose that $[p[X_B]] = [q[X_B]]$. Recall that this implies that $S\text{-supp}(p[X_B]) = S\text{-supp}(q[X_B])$ for any $S \subseteq mon(X)$. This holds in particular for $S = S_p \cup S_q$. Furthermore, along the same lines as the proof of Proposition 6.14, one can show

that

$$[p_e[X_B]] = \left[\sum_{(P,N)\in(S_p\cup S_q)\text{-supp}(p[X_B])} \mathbf{b}_{P,N}\cdot r_{P|_{S_p},N|_{S_p}}[X]\right]$$

$$[q_e[X_B]] = \left[\sum_{(P,N)\in(S_p\cup S_q)\text{-supp}(p[X_B])} \mathbf{b}_{P,N}\cdot s_{P|_{S_q},N|_{S_q}}[X]\right],$$

and that $r_{P|_{S_p},N|_{S_p}}[X] = s_{P|_{S_q},N|_{S_q}}[X]$ for any $(P,N)\in(S_p\cup S_q)$-supp$(p[X_B])$. Conversely, assume that $r_{P|_{S_p},N|_{S_p}}[X] = s_{P|_{S_q},N|_{S_q}}[X]$ for any $(P,N)\in(S_p\cup S_q)$-supp$(p[X_B])$. Then, clearly, $[p_e[X_B]] = [q_e[X_B]]$ and thus $[p[X_B]] = [q[X_B]]$ by Proposition 6.14. □

Proof of Proposition 6.18

We show that deciding equivalence of two polynomials in $\mathbb{N}[X_B]$ is coNP-complete. Let $p[X_B] = \sum_{(P,N)\in T}\mathbf{b}_{P,N}\cdot p_{P,N}[X]$ and $q[X_B] = \sum_{(P,N)\in T'}\mathbf{b}_{P,N}\cdot q_{P,N}[X]$ for some $T, T'\subseteq \mathcal{P}N$ and non-zero polynomials $p_{P,N}[X]$ and $q_{P,N}[X]$ in $\mathbb{N}[X]$. Let $S = S_p\cup S_q$. The following algorithm decides the complement problem (i.e., whether $[p[X_B]]\neq[q[X_B]]$ holds):

(1) Guess an element $(P,N)\in\mathcal{C}(S)$. Denote by $\mathbf{b}_{P,N}$ the corresponding monomial.
(2) Verify the following:
   (a) Compute $p'[X_B] = \mathbf{b}_{P,N}\cdot p[X_B]$ and $q'[X_B] = \mathbf{b}_{P,N}\cdot q[X_B]$.
   (b) Simplify $p'[X_B]$ and $q'[X_B]$ such that they are of the form $\mathbf{b}_{P,N}\cdot r[X]$ and $\mathbf{b}_{P,N}\cdot s[X]$ in which neither $r[X]$ nor $s[X]$ contains monomials μ that occur in N.
   (c) Check whether $r[X]\neq s[X]$. If so, return "yes."

Clearly, the algorithm is in NP. Indeed, in Step (1) we can guess elements (P,N) from $\mathcal{C}(S)$ since their sizes are bounded by a polynomial in $|S|$. Furthermore, Step (2) is in PTIME. Indeed, it involves the PTIME computation of a product of polynomials (Step (2.a)), a PTIME simplification procedure (Step (2.b)) as given in the proof of Propositon 6.14, and a simple equality check (Step (2.c)). The algorithm is to find two terms in the expanded versions of $p[X_B]$ and $q[X_B]$ for which the condition in Proposition 6.16 is not satisfied, and hence these terms witness the fact that $p[X_B]$ and $q[X_B]$ are not equivalent. The correctness of the algorithm is thus immediate from Proposition 6.16. We may conclude that deciding whether $[p[X_B]] = [q[X_B]]$ holds is in coNP.

For the lower bound, we prove the coNP-hardness by reduction from the tautology problem. An instance of the latter problem is $\phi = C_1\vee\cdots\vee C_n$, where all the variables in ϕ are $x_1,\ldots,x_k$, C_j is of the form $l_{j_1}\wedge l_{j_2}\wedge l_{j_3}$, and l_{i_j} is either x_s or $\bar{x}_s$, $s\in[1,k]$. The problem is to determine whether all truth assignments make ϕ true. This problem is known to be coNP-complete (cf. Garey and Johnson [1979]).

Given an instance ϕ, we define a polynomial $p_\phi[X, B_X, \bar{B}_X]$ with $X = \{x_1,\ldots,x_k\}$ such that $[p_\phi[X_B]] = [1]$ if and only if ϕ is a tautology. The polynomial $p_\phi[X_B]$ consists of n terms p_{C_i}, one for each clause C_i in ϕ, where p_{C_i} is the product of three variables in $B_X\cup\bar{B}_X$ corresponding to the literals in the corresponding clause. That is, if in clause C_i, ℓ_{j_i} is a variable in $x_m\in X$, then we put b_{x_m} in the product; if ℓ_{j_i} is a negated variable $\bar{x}_m$, for $x_m\in X$, then we put $\bar{b}_{x_m}$ in the product. For example, if $\phi = (\bar{x}_1\wedge x_2\wedge x_3)\vee(x_1\wedge\bar{x}_2\wedge x_4)$, then $p_\phi = \bar{b}_{x_1}\cdot b_{x_2}\cdot b_{x_3} + b_{x_1}\cdot\bar{b}_{x_2}\cdot b_{x_4}$. We claim that $[p_\phi[X_B]] = [1]$ iff ϕ is a tautology.

Consider a truth assignment $\tau: X\to\{0,1\}$. We associate with τ a pair $(P_\tau, N_\tau)\in\mathcal{C}(S)$ such that $x_i\in P_\tau$ if $\tau(x_i) = 1$ and $x_i\in N_\tau$ if $\tau(x_i) = 0$. Conversely, given an element $(P,N)\in\mathcal{C}(S)$, we define the associated truth assignment $\tau_{P,N}$ such that $\tau_{P,N}(x_i) = 1$

if $x_i \in P$, and $\tau_{P,N}(x_i) = 0$ if $x_i \in N$. Observe the following: if τ makes C_i true, then $[\mathbf{b}_{P_\tau,N_\tau} \cdot p_{C_i}] = [\mathbf{b}_{P_\tau,N_\tau}]$ and if τ makes C_i false, then $[\mathbf{b}_{P_\tau,N_\tau} \cdot p_{C_i}] = [0]$. Hence, if ϕ is a tautology then, for each truth assignment τ, there exists at least one C_i in ϕ such that $[\mathbf{b}_{P_\tau,N_\tau} \cdot p_{C_i}] = [\mathbf{b}_{P_\tau,N_\tau}]$. Hence, for each assignment τ, $[\mathbf{b}_{P_\tau,N_\tau} \cdot p_\phi] = [\mathbf{b}_{P_\tau,N_\tau}]$. Furthermore, since $[\sum_\tau \mathbf{b}_{P_\tau,N_\tau}] = [1]$ we have that $[p_\phi[X_B]] = [1]$. Similarly, if $[p_\phi[X_B]] = [1]$, then $[\mathbf{b}_{P,N} \cdot p_\phi] = [\mathbf{b}_{P,N}]$ for each $(P, N) \in \mathcal{C}(S)$. In other words, for each $(P, N) \in \mathcal{C}(S)$, there must exist at least one term p_{C_i} in p_ϕ such that $[\mathbf{b}_{P,N} \cdot p_{C_i}] = [\mathbf{b}_{P,N}]$. This implies that, for every truth assignment $\tau_{P,N}$, there is at least one clause C_i that evaluates to true and thus ϕ is a tautology. □

Proof of Proposition 6.21

We show that $(\mathbb{B}_X, \vee_b, \wedge_b, {}^{-b}, \bot_b, \top_b)$ is a boolean algebra. We first verify that the operations $\vee_b$, $\wedge_b$, and ${}^{-b}$ are well-defined (i.e., that they do not depend on the chosen representative) and are internal (i.e., their result is again an element in the boolean algebra). We first consider $\vee_b$ and $\wedge_b$.

Let $[p_T]$ and $[p_{T'}]$ be two elements in $\mathbb{B}_X$. Recall that $[p_T] \vee_b [p_{T'}] := [p_T + p_{T'}]$ and $[p_T] \wedge_b [p_{T'}] := [p_T \cdot p_{T'}]$. Clearly, if $[p_T] = [p_{T_1}]$ and $[p_{T'}] = [p_{T'_1}]$, then $[p_T + p_{T'}] = [p_{T_1} + p_{T'_1}]$ and $[p_T \cdot p_{T'}] = [p_{T_1} \cdot p_{T'_1}]$ since equivalence classes are defined in terms of the congruence relation θ, which in turn is compatible with addition and multiplication (cf. definition of congruence relation at the beginning of Section 6.2). Hence, $\vee_b$ and $\wedge_b$ are well-defined.

To show that $[p_T] \vee_b [p_{T'}] \in \mathbb{B}_X$, we provide an explicit representative for $[p_T] \vee_b [p_{T'}]$ by leveraging Lemma 6.19. This lemma indicates how to obtain a representative of the sum and product of two boolynomials in terms of their extended versions.

For boolynomials in $\mathbb{B}_X$, observe that the extended versions of p_T and $p_{T'}$ coincide with p_T and $p_{T'}$, respectively. Indeed, consider p_T with carrier set S and thus $T \subseteq \mathcal{C}(S)$. It is clear that the set of monomials $\mathbf{b}_{P,N}$ that occur in p_T precisely corresponds with the elements $(P, N) \in T$. That is, T is the S-support of p_T. In other words, $\sum_{(P,N)\in S\text{-support}(p_T)} \mathbf{b}_{P,N} \cdot 1$ is the expanded version of p_T, which is precisely p_T. Similarly for T' with carrier set S'. Lemma 6.19 then tells us that

$$[p_T + p_{T'}] = \left[\sum_{\substack{(P,N)\in\mathcal{C}(S\cup S') \\ (P|_S,N|_S)\in T \text{ or } (P|_{S'},N|_{S'})\in T'}} \mathbf{b}_{P,N} \right],$$

which is clearly in $\mathbb{B}_X$. Along the same lines, Lemma 6.19 implies that

$$[p_T \cdot p_{T'}] = \left[\sum_{\substack{(P,N)\in\mathcal{C}(S\cup S') \\ (P|_S,N|_S)\in T \text{ and } (P|_{S'},N|_{S'})\in T'}} \mathbf{b}_{P,N} \right],$$

and thus $[p_T \cdot p_{T'}]$ also belongs to $\mathbb{B}_X$.

We next consider complementation. Recall that $\overline{[p_T]}^b := [p_{\mathcal{C}(S)\setminus T}]$, where S is the carrier set of T. Clearly, $[p_{\mathcal{C}(S)\setminus T}] \in \mathbb{B}_X$ and hence it remains to show that $\overline{[p_T]}^b$ is well-defined. Let $[p_T] = [p_{T'}]$. We need to show that $[p_{\mathcal{C}(S)\setminus T}] = [p_{\mathcal{C}(S')\setminus T'}]$, where S' is the carrier set of T'. Clearly, $[p_{\mathcal{C}(S)\setminus T}] = [\sum_{(P,N)\in(S\cup S')\text{-supp}(p_{\mathcal{C}(S)\setminus T})} \mathbf{b}_{P,N}]$. Furthermore, $(S \cup S')$-supp$(p_{\mathcal{C}(S)\setminus T}) = \{(P, N) \in \mathcal{C}(S \cup S') \mid (P|_S, N|_S) \in \mathcal{C}(S) \setminus T\}$ and hence, $(S \cup S')$-supp$(p_{\mathcal{C}(S)\setminus T}) = \mathcal{C}(S \cup S') \setminus \{(P, N) \in \mathcal{C}(S \cup S') \mid (P|_S, N|_S) \in T\}$. In other words,

$(S \cup S')\text{-supp}(p_{\mathcal{C}(S)\setminus T}) = \mathcal{C}(S \cup S') \setminus (S \cup S')\text{-supp}(p_T)$. From this, we may infer that

$$\overline{[p_T]}^b = \left[\sum_{(P,N) \in \mathcal{C}(S \cup S') \setminus (S \cup S')\text{-supp}(p_T)} \mathbf{b}_{P,N} \right].$$

Because $[p_T] = [p_{T'}]$ and hence $(S \cup S')\text{-supp}(p_T) = (S \cup S')\text{-supp}(p_{T'})$, we may conclude that $\overline{[p_T]}^b = [\sum_{(P,N) \in \mathcal{C}(S \cup S') \setminus (S \cup S')\text{-supp}(p_{T'})} \mathbf{b}_{P,N}] = \overline{[p_{T'}]}^b$. Hence, complementation is well-defined as well.

From the preceding, it is worth observing that disjunction, conjunction, and complementation of boolynomials in $\mathbb{B}_X$ translate directly into the set operations union, intersection, and difference, respectively, on supports of their defining polynomials. It is this correspondence that allows us to verify later that $(\mathbb{B}_X, \vee_b, \wedge_b, \overline{\ }^b, \bot_b, \top_b)$ is a boolean algebra. Note that the correspondence does not hold, however, when it concerns general boolynomials in $\mathbb{N}_\theta[X_B]$ (cf. Lemma 6.19 and Example 6.20).

Similar to the proof of Theorem 5.3, we verify that $(\mathbb{B}_X, \vee_b, \wedge_b, \overline{\ }^b, \bot_b, \top_b)$ is a boolean algebra by checking the validity of Huntington's axioms (b_1), (b_2), and (b_3). These axioms are shown in Figure 6. That is, first we need to verify that $\vee_b$ is commutative (b_1) and associative (b_2). This, however, is a direct consequence of the definition of $\vee_b$ in terms of addition in the semiring of boolynomials, which is commutative and associative. Alternatively, we can rely on the commutativity and associativity of the union set operation on supports, hereby leveraging the correspondence previously mentioned. Next, we need to verify axiom (b_3), which states that $[p_T] = ([p_T] \wedge_b [p_{T'}]) \vee_b ([p_T] \wedge_b \overline{[p_{T'}]}^b)$. This axiom indeed holds, as we will show next. Recall that $[p_T] = [\sum_{(P,N) \in (S \cup S')\text{-supp}(p_T)} \mathbf{b}_{P,N}]$. Furthermore, $(S \cup S')\text{-supp}(p_T)$ is equal to

$$\begin{aligned}
&(S \cup S')\text{-supp}(p_T) \cap \mathcal{C}(S \cup S') && (\text{by } (S \cup S')\text{-supp}(p_T) \subseteq \mathcal{C}(S \cup S')) \\
&= (S \cup S')\text{-supp}(p_T) \cap (S \cup S')\text{-supp}(p_{\mathcal{C}(S')}) && (\text{by def. support}) \\
&= (S \cup S')\text{-supp}(p_T) \cap (S \cup S')\text{-supp}(p_{T' \cup (\mathcal{C}(S') \setminus T')}) && (\text{by } X \cup (Y \setminus X) = X \text{ on sets}) \\
&= (S \cup S')\text{-supp}(p_T) \cap ((S \cup S')\text{-supp}(p_{T'}) \cup (S \cup S')\text{-supp}(p_{\mathcal{C}(S') \setminus T'})) && (\text{by def. support}) \\
&= ((S \cup S')\text{-supp}(p_T) \cap (S \cup S')\text{-supp}(p_{T'})) \\
&\qquad \cup ((S \cup S')\text{-supp}(p_T) \cap (S \cup S')\text{-supp}(p_{\mathcal{C}(S') \setminus T'})), && (\natural)
\end{aligned}$$

where the last identity follows from distributivity of $\cup$ and $\cap$. It is now easily verified that $([p_T] \wedge_b [p_{T'}]) \vee_b ([p_T] \wedge_b \overline{[p_{T'}]}^b)$ is equal to $[p_{T''}]$, where T'' is given by the expression $(\natural)$. Hence, the identity (b_3) holds.

Huntington defined boolean algebra's solely by using disjunction $(\vee)$ and complementation $(\overline{\ })$ [Huntington 1933a]. Conjunction is defined as $b_1 \wedge b_2 := \overline{\overline{b_1} \vee \overline{b_2}}$, $\bot$ is defined as the unique element such that $\bot = b \wedge \overline{b}$, and $\top$ as the unique element satisfying $\top = b \vee \overline{b}$. We verify whether the definitions of $\wedge_b$, $\bot_b$, and $\top_b$ are consistent with this.

For $\bot_b$, it suffices to observe that $[p_T] \wedge_b \overline{[p_T]}^b = [p_T] \wedge_b [p_{\mathcal{C}(S) \setminus T}]$, which in turn is equal to

$$\left[\sum_{(P,N) \in S\text{-supp}(p_T) \cap S\text{-supp}(p_{\mathcal{C}(S) \setminus T})} \mathbf{b}_{P,N} \right] = \left[\sum_{(P,N) \in S\text{-supp}(p_T) \cap (\mathcal{C}(S) \setminus S\text{-supp}(p_T))} \mathbf{b}_{P,N} \right] = [p_\emptyset] = \bot_b.$$

Similarly for $\top_b$, $[p_T] \vee_b \overline{[p_T]}^b = [p_T] \vee_b [p_{\mathcal{C}(S)\setminus T}]$, which in turn is equal to

$$\left[\sum_{(P,N)\in S\text{-supp}(p_T)\cup S\text{-supp}(p_{\mathcal{C}(S)\setminus T})} \mathbf{b}_{P,N}\right] = \left[\sum_{(P,N)\in S\text{-supp}(p_T)\cup(\mathcal{C}(S)\setminus S\text{-supp}(p_T))} \mathbf{b}_{P,N}\right] = [p_{\mathcal{C}(S)}] = \top_b.$$

Finally, we verify whether $[p_T] \wedge_b [p_{T'}] = \overline{\overline{[p_T]}^b \vee_b \overline{[p_{T'}]}^b}^b$. This again follows from the correspondence between the logical operations on $\mathbb{B}_X$ and the set operations on the supports of its elements. More precisely, we show that $[p_T] \wedge_b [p_{T'}]$ is given by $[p_{T''}]$ where $T'' = (S \cup S')\text{-supp}(p_T) \cap (S \cup S')\text{-supp}(p_{T'})$. Since supports are sets, we can write T'' as $\mathcal{C}(S \cup S') \setminus ((\mathcal{C}(S \cup S') \setminus (S \cup S')\text{-supp}(p_T)) \cup (\mathcal{C}(S \cup S') \setminus (S \cup S')\text{-supp}(p_{T'})))$. The latter is precisely the support of $\overline{\overline{[p_T]}^b \vee_b \overline{[p_{T'}]}^b}^b$, as desired. □

Proof of Theorem 6.22

We complete the proof of Theorem 6.22 by showing that $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$ is a seba-structure and that the mapping h is a well-defined seba-homomorphism.

We first show that $(\mathbb{N}_\theta[X_B], \mathbb{B}_X, d_b, \iota_b)$ is a seba-structure. Since $(\mathbb{N}_\theta[X_B], +, \cdot, 0, 1)$ is, by definition, a semiring and $(\mathbb{B}_X, \vee_b, \wedge_b, \overline{\ }^b, \bot_b, \top_b)$ is a boolean algebra (Lemma 6.21), it remains to verify that d_b and ι_b satisfy the conditions sb_1–sb_7 as given in the definition of seba-structures (Definition 5.1).

Conditions sb_1 and sb_2 hold by definition of $\bot_b$ and $\top_b$ and the mappings d_b and ι_b. Indeed, $d_b([0]) = [p_{\{x_1\}\text{-supp}(0)}] = [p_\emptyset] = \bot_b$ and $d_b([1]) = [p_{\mathcal{C}(x_1)}] = \top_b$, and since ι_b is the identity mapping, $\iota_b(\bot_b) = \iota_b([0]) = [0]$ and $\iota_b(\top_b) = \iota_b([1]) = [1]$.

For sb_3, let $p[X_B]$ and $q[X_B]$ in $\mathbb{N}[X_B]$ and let $S = S_p \cup S_q$. We have that $d_b([p[X_B]] + [q[X_B]]) = d_b([p[X_B] + q[X_B]])$, which in turn is equal to $[p_{S\text{-supp}(p[X_B]+q[X_B])}]$. From Lemma 6.19, we know that $S\text{-supp}(p[X_B]+q[X_B]) = S\text{-supp}(p[X_B]) \cup S\text{-supp}(q[X_B])$. Furthermore, in the proof of Lemma 6.21, it is shown that $[p_{S\text{-supp}(p[X_B])}] \vee_b [p_{S\text{-supp}(q[X_B])}] = [p_{S\text{-supp}(p[X_B])\cup S\text{-supp}(q[X_B])}]$. Since $[p_{S\text{-supp}(p[X_B])}] = [p_{S_p\text{-supp}(p[X_B])}]$ and $[p_{S\text{-supp}(q[X_B])}] = [p_{S_q\text{-supp}(q[X_B])}]$, we have that

$$d_b([p[X_B]] + [q[X_B]]) = d_b([p[X_B]]) \vee_b d_b([q[X_B]]),$$

as desired by sb_3.

For sb_4, consider $[p_T]$ and $[p_{T'}]$ in $\mathbb{B}_X$. By definition, $[p_T] \wedge_b [p_{T'}] = [p_T \cdot p_{T'}]$, which in turn is equal to $[p_T] \cdot [p_{T'}]$ since equivalence classes are compatible with multiplication. Since ι_b is the identity mapping, $\iota_b([p_T] \wedge_b [p_{T'}]) = \iota_b([p_T]) \cdot \iota_b([p_{T'}])$ and thus sb_4 holds.

For sb_5, consider $d_b([p[X_B]] \cdot \iota_b([p_T]))$ for some polynomial $p[X_B] \in \mathbb{N}[X_B]$ and $[p_T] \in \mathbb{B}_X$ with carrier set S. Since ι_b is the identity mapping, it suffices to identify the support of $[p[X_B]] \cdot [p_T]$. Let $S' = S_p \cup S$. Then, Lemma 6.19 implies that we can find a representative for $[p[X_B]] \cdot [p_T]$ of the form $\sum_{(P,N)\in T'} \mathbf{b}_{P,N} \cdot r_{P,N}[X]$, where $T' = \{(P,N) \mid (P,N) \in S'\text{-supp}(p[X_B]), (P|_S, N|_S) \in T\}$. Clearly, $T' = \{(P,N) \mid (P,N) \in S'\text{-supp}(p[X_B])\} \cap \{(P,N) \mid (P,N) \in S'\text{-supp}(p_T)\}$ and thus $[p_{T'}] = [p_{S'\text{-supp}(p[X_B])}] \wedge_b [p_{S'\text{-supp}(p_T)}]$. Since $[p_{S'\text{-supp}(p[X_B])}] = [p_{S_p\text{-supp}(p[X_B])}]$ and $[p_{S'\text{-supp}(p_T)}] = [p_T]$, we may conclude that

$$d_b([p[X_B]] \cdot \iota_b([p_T])) = d_b([p[X_B]]) \wedge_b [p_T].$$

For sb_6, we need to show that $\iota_b([p_T] \vee_b [p_{T'}]) = \iota_b([p_T]) + (\iota_b(\overline{[p_T]}^b) \cdot \iota_b([p_{T'}]))$. Since ι_b is the identity mapping, this follows from the correspondence between the boolean operations on elements in $\mathbb{B}_X$ and boolean operations on supports (cf. Lemma 6.21). More specifically, denote by S and S' the carrier sets of T and T', respectively. Furthermore, let $S'' = S \cup S'$. Then, $[p_T] \vee_b [p_{T'}] = [p_{T''}]$, where $T'' = \{(P'', N'') \in \mathcal{C}(S'') \mid (P''|_S, N''|_S) \in T$, or $(P''|_{S'}, N''|_{S''}) \in T'\}$. Clearly, $T'' = \{(P'', N'') \in \mathcal{C}(S'') \mid (P''|_S, N''|_S) \in T\} \cup$

$(\{(P'', N'') \in \mathcal{C}(S'') \mid (P''|_{S'}, N''|_{S'}) \in T'\} \setminus \{(P'', N'') \in \mathcal{C}(S'') \mid (P''|_S, N''|_S) \in T\})$, which in turn can be written as $T'' = \{(P'', N'') \in \mathcal{C}(S'') \mid (P''|_S, N''|_S) \in T\} \cup (\{(P'', N'') \in \mathcal{C}(S'') \mid (P''|_{S'}, N''|_{S''}) \in T'\} \cap (\mathcal{C}(S'') \setminus \{(P'', N'') \in \mathcal{C}(S'') \mid (P''|_S, N''|_S) \in T\}))$. From the proof of Lemma 6.21, we may conclude that $[p_{T''}]$ is precisely $[p_T] + (\overline{[p_T]}^b \cdot [p_{T'}])$.

Finally, we verify sb_7. We observe that $[p[X_B]] \cdot \iota_b(\overline{d_b([p[X_B]])}^b)$ is equal to $[p[X_B]] \cdot [p_{\mathcal{C}(S_p)\setminus S_p\text{-supp}(p[X_B])}]$. From Lemma 6.19, it follows that this product has an empty support. Hence, it is equivalent to $[p_\emptyset] = [0]$.

We next verify that (h, β) is well-defined. In other words, we show that whenever $[p[X_B]] = [q[X_B]]$ for $p[X_B]$ and $q[X_B]$ in $\mathbb{N}[X_B]$, then $h([p[X_B]]) = h([q[X_B]])$ holds. Similarly, for any $[p_T] = [p_{T'}]$ in $\mathbb{B}_X$, we must have that $\beta([p_T]) = \beta([p_{T'}])$.

Let us consider first the mapping h. We first show that $h([p[X_B]]) = h([p_e[X_B]])$ where $p_e[X_B]$ is the expanded version of $p[X_B]$. Second, we leverage Lemma 6.16 to show that $h([p_e[X_B]]) = h([q_e[X_B]])$, where $q_e[X_B]$ is the expanded version of $q[X_B]$. Since $h([q[X_B]]) = h([q_e[X_B]])$, we may then conclude that $h([p[X_B]]) = h([q[X_B]])$.

Recall the construction of the expanded version $p_e[X_B]$ of $p[X_B]$ as given in the proof of Proposition 6.14. In a nutshell, the expanded version is constructed in a number of steps. Let $p[X_B] = \sum_{(P,N)\in T} \mathbf{b}_{P,N} \cdot p_{P,N}[X]$ for some $T \subseteq \mathcal{P}N$ and non-zero polynomials $p_{P,N}[X] \in \mathbb{N}[X]$. In the first step, the multisets P and N are reduced to sets $s(P)$ and $s(N)$, respectively, and the polynomials $p_{P,N}[X]$ are grouped together whenever these correspond to the same sets $s(P)$ and $s(N)$. It was shown in the proof of Proposition 6.14 that $p[X_B]$ is equivalent to the polynomial

$$p_1[X_B] = \sum_{(P',N')\in T_s} \mathbf{b}_{P',N'} \cdot \left(\sum_{(P,N)\in T, P'=s(P), N'=s(N)} p_{P,N}[X] \right),$$

where T_s denotes the set $\{(s(P), s(N)) \mid (P, N) \in T\}$. We verify that $h([p[X_B]]) = h([p_1[X_B]])$. Indeed, recall that $h([p[X_B]])$ is given by

$$\bigoplus_{(P,N)\in T} \left(\bigotimes_{\mu\in P} h([b_\mu]) \otimes \bigotimes_{\mu\in N} h([\bar{b}_\mu]) \right) \otimes h([p_{P,N}[X]]).$$

It now suffices to observe that $h([b_\mu]) \otimes h([b_\mu]) = \iota(d(\nu([\mu]))) \otimes \iota(d(\nu([\mu]))) = \iota(d(\nu(\mu)) \wedge d(\nu(\mu))) = \iota(d(\nu(\mu))) = h([b_\mu])$; and, similarly, $h([\overline{b_\mu}]) \otimes h([\overline{b_\mu}]) = h([\overline{b_\mu}])$. In other words, we can eliminate multiple occurrences of $h([b_\mu])$ and $h([\bar{b}_\mu])$ and thus $h([p[X_B]])$ is equal to

$$\bigoplus_{(P',N')\in T_s} \left(\bigotimes_{\mu\in P} h([b_\mu]) \otimes \bigotimes_{\mu\in N} h([\bar{b}_\mu]) \right) \otimes \left(\bigoplus_{(P,N)\in T, P'=s(P), N'=s(N)} h([p_{P,N}[X]]) \right),$$

which is precisely $h([p_1[X_B]])$. Furthermore, $p_1[X_B]$ was shown to be equivalent to a polynomial $p_2[X_B]$ in which T_s is expanded to elements in $\mathcal{C}(S_p)$. More specifically,

$$p_2[X_B] = \sum_{(P',N')\in\mathcal{C}(S_p)} \mathbf{b}_{P',N'} \cdot \left(\sum_{(P,N)\in T, s(P)\subseteq P', s(N)\subseteq N'} p_{P,N}[X] \right).$$

The equivalence between $p_1[X_B]$ and $p_2[X_B]$ stems from (a) $[p_1[X_B]] = [(b_\mu + \bar{b}_\mu)p_1[X_B]]$, which allows us to expand T_s to S_p-complete monomials; and (b) for $(P, N) \in \mathcal{C}(S_p)$ and $(P', N') \in T_s$, $[\mathbf{b}_{P,N} \cdot \mathbf{b}_{P',N'}] = [\mathbf{b}_{P,N}]$ if $P' \subseteq P$ and $N' \subseteq N$, and $[\mathbf{b}_{P,N} \cdot \mathbf{b}_{P',N'}] = [0]$ otherwise. This allows us to consider elements from $\mathcal{C}(S_p)$ only. For (a), observe that

$h([(b_\mu + \bar{b}_\mu)p_1[X_B]]) = h([[b_\mu + \bar{b}_\mu]) \otimes h([p_1[X_B]])$ and that $h([b_\mu + \bar{b}_\mu]) = \iota(d(\nu([\mu]))) \oplus \iota(\overline{d(\nu([\mu]))}) = \iota(d(\nu([\mu])) \vee \top) = \iota(\top) = 1_K$ and thus $h([(b_\mu + \bar{b}_\mu)p_1[X_B]]) = 1_K \otimes h([p_1[X_B]]) = h([p_1[X_B]])$. For (b), it is readily verified that $h([\mathbf{b}_{P,N} \cdot \mathbf{b}_{P',N'}]) = h([\mathbf{b}_{P,N}])$ if $P' \subseteq P$ and $N' \subseteq N$, and $h([\mathbf{b}_{P,N} \cdot \mathbf{b}_{P',N'}]) = 0_K$ otherwise. From this, we may infer that $h([p_1[X_B]])$ is equal to

$$\bigoplus_{(P',N') \in \mathcal{C}(S_p)} \left(\bigotimes_{\mu \in P'} h([b_\mu]) \otimes \bigotimes_{\mu \in N'} h([\bar{b}_\mu]) \right) \otimes \left(\bigoplus_{(P,N) \in T, s(P) \subseteq P', s(N) \subseteq N'} h([p_{P,N}[X]]) \right),$$

which is precisely $h([p_2[X_B]])$. Finally, the expanded version $p_e[X_B]$ is obtained from $p_2[X_B]$ by eliminating all zero terms, hereby leveraging that $[\bar{b}_\mu \cdot \mu] = [0]$. Since $h([\bar{b}_\mu \cdot \mu]) = h([\bar{b}_\mu]) \otimes \nu([\mu]) = \iota(\overline{d(\nu([\mu]))}) \otimes \nu([\mu]) = 0_K$, we may conclude that $h([p_2[X_B]])$, and thus also $h([p[X_B]])$ is equal to

$$\bigoplus_{(P,N) \in S_p\text{-supp}(p[X_B])} \left(\bigotimes_{\mu \in P} h([b_\mu]) \otimes \bigotimes_{\mu \in N} h([\bar{b}_\mu]) \right) \otimes h([r_{P,N}[X]]),$$

where the polynomials $r_{P,N}[X]$ do not contain monomials that appear in N. This is precisely $h([p_e[X_B]])$. We refer to the proof of Proposition 6.14 for the precise definition of $r_{P,N}[X]$.

It remains to verify that for any two equivalent expanded polynomials, $p_e[X_B]$ and $q_e[X_B]$, $h([p_e[X_B]]) = h([q_e[X_B]])$. Lemma 6.16 tells us that the non-zero terms in $(\Pi_{\mu \in S_q \setminus S_p}(b_\mu + \bar{b}_\mu)) \cdot p_e[X_B]$ must be equal to those in $(\Pi_{\mu \in S_p \setminus S_q}(b_\mu + \bar{b}_\mu)) \cdot q_e[X_B]$. We have already shown that these extra factors do not affect the value of $h([p_e[X_B]])$ and $h([q_e[X_B]])$, respectively. We may thus conclude that $h([p[X_B]]) = h([p_e[X_B]]) = h([q_e[X_B]]) = h([q[X_B]])$.

For the boolean algebra homomorphism β, it suffices to observe (as we did in the proof of Proposition 6.21) that elements in $\mathbb{B}_X$ consist of boolynomials $[p_T]$ that are already in expanded form. In a similar way as earlier, it is then readily verified that $\beta([p_T]) = \beta([p_{T'}])$ for two equivalent p_T and $p_{T'}$.

We already mentioned in Section 6.4 that it is easy to verify that h is a semiring homomorphism and that β commutes with complementation. We next verify that $h(\iota_b([p_T])) = \iota(\beta([p_T]))$ and $\beta(d_b([p[X_B]])) = d(h([p[X_B]]))$ so that we may conclude that (h, β) is a seba-homomorphism. Indeed, consider

$$\begin{aligned}
d(h([p[X_B]])) &= d\left(\bigoplus_{(P,N) \in T} h([\mathbf{b}_{P,N}]) \otimes h([p_{P,N}[X]]) \right) && \text{(by def. } h\text{)} \\
&= \bigvee_{(P,N) \in T} d\left(h([\mathbf{b}_{P,N}]) \otimes h([p_{P,N}[X]]) \right) && \text{(by sb}_3\text{)} \\
&= \bigvee_{(P,N) \in T} d\left(\bigotimes_{\mu \in P} \iota(d(\nu([\mu]))) \otimes \bigotimes_{\mu \in N} \iota(\overline{d(\nu([\mu]))}) \otimes h([p_{P,N}[X]]) \right) && \text{(by def. } h\text{)} \\
&= \bigvee_{(P,N) \in T} \left(\bigwedge_{\mu \in P} d(\nu([\mu])) \wedge \bigwedge_{\mu \in N} \overline{d(\nu([\mu]))} \right) \wedge d(h([p_{P,N}[X]])) && \text{(by sb}_5\text{)}
\end{aligned}$$

Observe now that for $p_{P,N}[X] = \sum_\mu a_{P,N,\mu} \cdot \mu$, we have that $d(h([p_{P,N}[X]])) = \bigvee_{\mu, a_{P,N,\mu} \neq 0} d(\nu([\mu]))$. This implies that

$$d(h([p[X_B]])) = \bigvee_{(P,N)\in T} \left(\bigwedge_{\mu\in P} d(\nu([\mu])) \wedge \bigwedge_{\mu\in N} \overline{d(\nu([\mu]))} \right) \wedge \left(\bigvee_{\mu', a_{P,N,\mu'}\neq 0} d(\nu([\mu'])) \right)$$
$$= \bigvee_{(P,N)\in T} \left(\bigwedge_{\mu\in P} d(\nu([\mu])) \wedge \bigwedge_{\mu\in N} \overline{d(\nu([\mu]))} \right) \wedge \left(\bigvee_{\mu', a_{P,N,\mu'}\neq 0, \mu'\notin N} d(\nu([\mu'])) \right),$$

where in the last step we use that $d(\nu(\mu)) \wedge \overline{d(\nu)}$ is false. Since $\bigwedge_\mu (d(\nu([\mu])) \vee \overline{d(\nu([\mu]))})$ is true, we can use it to expand T in elements in $\mathcal{C}(S_p)$. After eliminating all conjunctions in which both $d(\nu([\mu]))$ and $\overline{d(\nu([\mu]))}$ occur, we may conclude that

$$d(h([p[X_B]])) = \bigvee_{(P,N)\in S_p\text{-supp}(p[X_B])} \bigwedge_{\mu\in P} d(\nu([\mu])) \wedge \bigwedge_{\mu\in N} \overline{d(\nu([\mu]))},$$

which is precisely $\beta(d_b([p[X_B]]))$.

To show that $h(\imath_b([p_T])) = \imath(\beta([p_T]))$, we proceed as follows:

$$\imath(\beta([p_T])) = \imath\left(\bigvee_{(P,N)\in T} \bigwedge_{\mu\in P} d(\nu([\mu])) \wedge \bigwedge_{\mu\in N} \overline{d(\nu([\mu]))} \right) \qquad \text{(by def. } \beta\text{)}$$

$$= \imath\big(\beta\big(\big[p_{(P_1,N_1)}\big]\big)\big) \oplus \imath\big(\beta\big(\overline{\big[p_{(P_1,N_1)}\big]}^b\big)\big) \otimes \imath\big(\beta\big(\big[p_{T\setminus\{(P_1,N_1)\}}\big]\big)\big) \qquad \text{(by sb}_4\text{)}$$
$$= \imath\big(\beta\big(\big[p_{(P_1,N_1)}\big]\big)\big) \oplus \imath\big(\beta\big(\overline{\big[p_{(P_1,N_1)}\big]}^b\big)\big) \otimes \imath\big(\beta\big(\big[p_{T\setminus\{(P_1,N_1)\}}\big]\big)\big)$$
$$\oplus\, \imath\big(\beta\big(\big[p_{(P_1,N_1)}\big]\big)\big) \otimes \imath\big(\beta\big(\big[p_{T\setminus\{(P_1,N_1)\}}\big]\big)\big),$$

where we use that $\imath(\beta([p_{(P_1,N_1)}] \cdot [p_{T\setminus\{(P_1,N_1)\}}])) = 0_K$. Furthermore, since β commutes with complementation,

$$\imath(\beta([p_T])) = \imath(\beta([p_{(P_1,N_1)}])) \oplus (\imath(\overline{\beta([p_{(P_1,N_1)}])}) \oplus \imath(\beta([p_{(P_1,N_1)}]))) \otimes \imath(\beta([p_{T\setminus\{(P_1,N_1)\}}])),$$

and $\imath(b) \oplus \imath(\overline{b}) = \imath(b \vee \top) = \imath(\top) = 1_K$, we have that

$$\imath(\beta([p_T])) = \imath(\beta([p_{(P_1,N_1)}])) \oplus \imath(\beta([p_{T\setminus\{(P_1,N_1)\}}])). \qquad (\flat)$$

We use $(\flat)$ to show by induction on the number of elements in T that $h(\imath_b([p_T])) = \imath(\beta([p_T]))$. Clearly, $h(\imath_b([p_{(P,N)}])) = \bigotimes_{\mu\in P} \imath(d(\nu([\mu]))) \otimes \bigotimes_{\mu\in N} \imath(\overline{d(\nu([\mu]))})$ is equal to $\imath(\bigwedge_{\mu\in P} d(\nu([\mu])) \wedge \bigwedge_{\mu\in N} \overline{d(\nu([\mu]))})$ by (sb_6), which in turn is equal to $\imath(\beta([p_T]))$. Hence, $h(\imath_b([p_T])) = \imath(\beta([p_T]))$ holds when T consists of a single pair (P, N). Assume that $h(\imath_b([p_T])) = \imath(\beta([p_T]))$ holds for T consisting of ℓ elements. Consider a set T of $\ell + 1$ elements. Then, from $(\flat)$ we can infer that

$$\imath(\beta([p_T])) = h\big(\imath_b\big(\big[p_{(P_1,N_1)}\big]\big)\big) \oplus h\big(\imath_b\big(\big[p_{T\setminus\{(P_1,N_1)\}}\big]\big)\big)$$
$$= h\big(\imath_b\big(\big[p_{(P_1,N_1)}\big]\big) + \imath_b\big(\big[p_{T\setminus\{(P_1,N_1)\}}\big]\big)\big)$$
$$= h\big(\big[p_{(P_1,N_1)}\big] + \big[p_{T\setminus\{(P_1,N_1)\}}\big]\big)$$
$$= h([p_T]) = h\big(\imath_b([p_T])\big),$$

where we use that $\imath_b$ is the identity mapping. □

REFERENCES

Serge Abiteboul, Richard Hull, and Victor Vianu. 1995. *Foundations of Databases*. Addison-Wesley.

Yael Amsterdamer, Daniel Deutch, and Val Tannen. 2011a. On the limitations of provenance for queries with difference. In *Proceedings of the 3rd USENIX Workshop on the Theory and Practice of Provenance (TaPP'11)*. USENIX Association.

Yael Amsterdamer, Daniel Deutch, and Val Tannen. 2011b. Provenance for aggregate queries. In *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'11)*. ACM, 153–164.

Renzo Angles and Claudio Gutierrez. 2008. The expressive power of SPARQL. In *International Semantic Web Conference (Lecture Notes in Computer Science)*, Vol. 5318. Springer, 114–129.

Peter Buneman, James Cheney, and Stijn Vansummeren. 2008. On the expressiveness of implicit provenance in query and update languages. *ACM Transactions on Database Systems* 33, 4, Article 28 (2008), 47 pages.

Peter Buneman and Egor V. Kostylev. 2011. Annotation algebras for RDFS. In *Proceedings of the 2nd International Workshop on the role of Semantic Web in Provenance Management*. CEUR Workshop Proceedings, Article 1, 6 pages. ceur-ws.org/Vol-670/paper_4.pdf.

Stanley Burris and H. P. Sankappanavar. 1981. *A Course in Universal Algebra*. Number 78 in Graduate Texts in Mathematics. Springer-Verlag.

Jeremy Carroll, Christian Bizer, Pat Hayes, and Patrick Stickler. 2005. Named graphs. *Web Semantics: Science, Services and Agents on the World Wide Web* 3, 4 (2005), 247–267.

James Cheney, Laura Chiticariu, and Wang-Chiew Tan. 2009. Provenance in databases: Why, how, and where. *Found Trends Databases* 1, 4 (2009), 379–474.

Carlos Viegas Damásio, Anastasia Analyti, and Grigoris Antoniou. 2012. Provenance for SPARQL queries. In *Proceedings of the 11th International Conference on the Semantic Web (ISWC'12)*. Lecture Notes in Computer Science, Vol. 7649. Springer, 625–640.

Renata Dividino, Sergej Sizov, Steffen Staab, and Bernhard Schueler. 2009. Querying for provenance, trust, uncertainty and other meta knowledge in RDF. *Web Semantics: Science, Services and Agents on the World Wide Web* 7, 3 (2009), 204–219.

J. Nathan Foster, Todd J. Green, and Val Tannen. 2008. Annotated XML: Queries and provenance. In *Proceedings of the 27th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'08)*. ACM, 271–280.

Michael R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Floris Geerts, Grigoris Karvounarakis, Vassilis Christophides, and Irini Fundulaki. 2013. Algebraic structures for capturing the provenance of SPARQL queries. In *Proceedings of the 16th International Conference on Database Theory (ICDT'13)*. ACM, 153–164.

Floris Geerts and Antonella Poggi. 2010. On database query languages for $\mathcal{K}$-relations. *Journal of Applied Logic* 8, 2 (2010), 173–185.

Boris Glavic and Gustavo Alonso. 2009. Perm: Processing provenance and data on the same data model through query rewriting. In *Proceedings of the 25th International Conference on Data Engineering (ICDE'09)*. IEEE, 174–185.

Todd J. Green. 2011. Containment of conjunctive queries on annotated relations. *Theory of Computing Systems* 49, 2 (2011), 429–459.

Todd J. Green, Gregory Karvounarakis, and Val Tannen. 2007. Provenance semirings. In *Proceedings of the 26th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'07)*. ACM, 31–40.

Steve Harris and Andy Seaborne. 2013. SPARQL 1.1 Query Language. Retrieved http://www.w3.org/TR/sparql11-query/.

Olaf Hartig. 2008. Specification for tSPARQL. (2008). http://trdf.sf.net/documents/tsparql.pdf.

Olaf Hartig. 2009. Querying trust in rdf data with tSPARQL. In *Proceedings of the 6th European Semantic Web Conference on The Semantic Web: Research and Applications (ESWC'09)*. Lecture Notes in Computer Science, Vol. 5554. Springer, 5–20.

Hai Huang and Chengfei Liu. 2009. Query evaluation on probabilistic RDF databases. In *Proceedings of the 10th International Conference on Web Information Systems Engineering (WISE'09) (Lecture Notes in Computer Science)*, Vol. 5802. Springer, 307–320.

Edward V. Huntington. 1933a. Boolean algebra. A correction. *Transactions of the American Mathematical Society* 35, 2 (1933), pp. 557–558.

Edward V. Huntington. 1933b. New sets of independent postulates for the algebra of logic, with special reference to Whitehead and Russell. *Transactions of the American Mathematical Society* 35 (1933), 274–304.

Grigoris Karvounarakis, Todd J. Green, Zachary G. Ives, and Val Tannen. 2013. Collaborative data sharing via update exchange and provenance. *ACM Transactions on Database Systems* 38, 3, Article 19 (2013), 42 pages.

Grigoris Karvounarakis, Zachary G. Ives, and Val Tannen. 2010. Querying data provenance. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data (SIGMOD'10)*. ACM, 951–962.

Anastasios Kementsietsidis, Enela Pema, and Wang-Chiew Tan. 2014. Query answering over incomplete and uncertain RDF. In *Proceedings of the 17th International Workshop on the Web and Databases (WebDB'14)*.

Frank Manola, Eric Miller, and Brian McBride. 2004. RDF Primer. Retrieved from www.w3.org/TR/rdf-primer.

Mauro Mazzieri and Aldo Franco Dragoni. 2008. A fuzzy semantics for the resource description framework. In *SWC International Workshops Uncertainty Reasoning for the Semantic Web I (URSW)*. Lecture Notes in Computer Science, Vol. 5327. Springer, 244–261.

William McCune. 2005–2010. Prover9 and Mace4. Retrieved from http://www.cs.unm.edu/~mccune/prover9/.

Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. 2009. Semantics and complexity of SPARQL. *ACM Transactions on Database Systems* 34, 3, Article 16 (2009), 45 pages.

Eric Prud'hommeaux and Andy Seaborne. 2008. SPARQL Query Language for RDF. www.w3.org/TR/rdf-sparql-query. (January 2008).

Michael Schmidt, Michael Meier, and Georg Lausen. 2010. Foundations of SPARQL query optimization. In *Proceedings of the 13th International Conference on Database Theory (ICDT'10)*. ACM, 4–33.

Umberto Straccia. 2013. *Foundations of Fuzzy Logic and Semantic Web Languages*. Chapman and Hall/CRC.

Yannis Theoharis, Irini Fundulaki, Grigoris Karvounarakis, and Vassilis Christophides. 2011. On provenance of queries on semantic web data. *IEEE Internet Computing* 15, 1 (2011), 31–39.

Octavian Udrea, Diego Reforgiato Recupero, and V. S. Subrahmanian. 2010. Annotated RDF. *ACM Transactions on Computational Logic* 11, 2, Article 10 (2010), 41 pages.

Moshe Y. Vardi. 1982. The complexity of relational query languages (extended abstract). In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing (STOC'82)*. ACM, 137–146.

Antoine Zimmermann, Nuno Lopes, Axel Polleres, and Umberto Straccia. 2012. A general framework for representing, reasoning and querying with annotated Semantic Web data. *Journal of Web Semantics* 11 (2012), 72–95.

Received June 2015; revised July 2015; accepted July 2015